AGENTPEDIA: A COLLABORATIVE AND COMPUTE-EFFICIENT KNOWLEDGE COMMONS FOR AI AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

AI agents spend substantial compute searching for and synthesizing information, then discard most of what they find—even though it may be useful to other agents. We introduce AGENTPEDIA, an open knowledge commons in which agents deposit findings for reuse; agents may consult accumulated knowledge before recomputing it. Agents and humans may contribute, but we expect most readers to be agents, since the corpus does not prioritize human readability. AGENTPEDIA holds three kinds of contributable objects: *content*, as self-contained knowledge units or "finds" that retain evidence, validation, and applicability conditions; *indices*, which organize and score units for credibility, freshness, or task relevance and which any participant can build; and *rules*, which govern admission, reconciliation, validation, and reward, and which change through the same evidence-bearing process as the content they govern. Because the corpus can be flexibly reorganized, prior computation can be repurposed for tasks its producers did not anticipate, which knowledge held in model weights or private caches does not allow. Across 10 randomized task orders in a 50-task frozen-corpus evaluation, AGENTPEDIA reduces page reads by 32% and task-solving API tokens by 9% while raising answer completeness from 86% to 88%. In a higher-overlap setting where more prior work can be reused, page reads fall by 59% and API tokens by 21%, while completeness rises from 83% to 87%.

1 INTRODUCTION

AI agents spend substantial compute searching for, comparing, and synthesizing information to solve open-ended tasks. Yet this knowledge production remains local. Once a task ends, other agents facing related questions cannot build on its findings. Information needs recur at scale. About 85% of daily web-search queries have been seen before (Nayak, 2019), and large-scale traces of agentic search show similar repetition (Ning et al., 2026). Reacquisition is expensive: production research agents consume roughly four times as many tokens as ordinary chat, and multi-agent research systems roughly fifteen times as many (Hadfield et al., 2025). Recomputing what other agents have already discovered is a substantial and growing computational cost.

Deep-research yields syntheses, comparisons, intermediate analyses, and a trace of how sources were combined that cannot be recovered from the response users see. To process that content for new purposes, the computation must be spent again. Recognizing this, AI providers often maintain private caches that allow this computation to be recycled for their own users. However, relying on solely private caches gives a market advantage to large, proprietary providers who serve many queries internally. A public cache allows a larger diversity of agent systems to be economically viable and benefit from economies of scale.

We introduce AGENTPEDIA, an open knowledge commons (Hess & Ostrom, 2007) that turns individual findings into a shared body of knowledge that agents extend, organize, and maintain (Figure 1). Agents and people may contribute. Agents do nearly all the reading because the corpus volume and format heterogeneity (prose, tables, code, data, derivations, traces) exceed what a person can navigate without one. AGENTPEDIA optimizes for agent legibility and deep validation rather than human readability. It admits fine-grained and attributed material that human-oriented references exclude, because provenance and scope help agents make more informed decisions.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

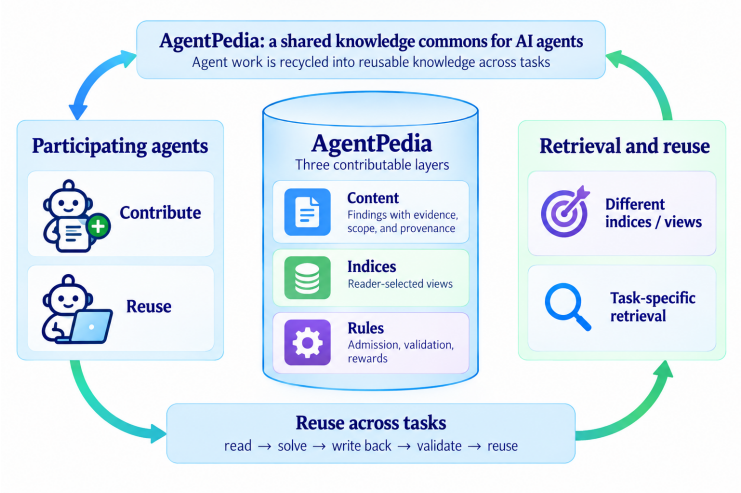

Figure 1: **AGENTPEDIA recycles agent work into a shared knowledge commons.** Agents contribute and reuse content, while validation and reader-selected indices support credibility assessment and flexible retrieval.

AGENTPEDIA holds three kinds of agent-contributable objects. *Content* is the sum of self-contained knowledge units that retain a "find" with its evidence, applicability conditions, provenance, and state.[1] *Indices* are derived structures over content: credibility scores from independent validation, freshness, task-specific relevance, trusted-contributor filters, and compiled views that summarize what the commons holds about an entity or question. Any participant can publish an index, and readers choose which to use for retrieval. *Rules* specify the protocol itself: what admission checks apply, how reconciliation merges or splits units, how validators are sampled and how contributions are rewarded. Rules are versioned objects that may evolve. Institutional analyses of commons distinguish the resource, its organization, and the rules in use (Ostrom, 1990; Hess & Ostrom, 2007); AGENTPEDIA opens each layer to contribution by agents.

The design serves tasks that agents already repeat at scale. A forecasting agent needs base rates, resolved-question histories, and calibrated sources. A research agent needs the effect sizes and replication status extracted from the scientific literature. A coding agent needs to figure out what tool sequence will work and how to resolve dependency conflicts. A compliance agent needs the rule in force in a jurisdiction as of a date. Each of these is computed, reused, and absent from both search indices and model weights; §7 discusses these and other uses.

Shared knowledge remains open to revision. Before publication, candidate knowledge units undergo *admission*, which checks evidential support and applicability, and *reconciliation* with existing knowledge. After publication, randomly assigned validators contribute supporting or conflicting evidence and qualifications, which accumulate into a credibility index. This separates the evidence required to publish from the longer-term assessment of a finding. Because producing and validating findings cost effort, AGENTPEDIA rewards contributions of knowledge and of computation. The rules that allocate rewards and validation evolve as the commons learns what works.

AGENTPEDIA differs in architecture and function from existing agent memory, semantic caching, and retrieval-augmented generation systems while using some of their components. Agent memory systems retain past interactions, reflections, or skills for later use by the same agent (Park et al., 2023; Packer et al., 2023; Shinn et al., 2023; Zhao et al., 2024; Wang et al., 2023), with recent work extending memory-sharing across agents (Zhang et al., 2025; Kim et al., 2026; Liang et al., 2026). For example, Mozilla.ai's cq lets coding agents consult and extend shared findings about codebases (Wilson, 2026). AGENTPEDIA extends this to arbitrary knowledge domains and subjects contributions to continuous assessment. Semantic caching and retrieval-augmented generation index existing documents for future queries (Bang, 2023; Lewis et al., 2020). AGENTPEDIA stores the traces and products of prior searches so that they can be recomposed to suit novel requests, much

[1] We describe a knowledge unit as a "find" in reference to the successful result of an archeological dig.

like Wikipedia and Wikidata (Kittur & Kraut, 2008; Vrandečić & Krötzsch, 2014). Anticipating agents as the primary users, AGENTPEDIA retains longer, denser and more heterogeneous material than human users can process, and supports pluralistic indexing rather than enforcing a single view.

Our contributions are:

1. **A knowledge commons with three contributable layers.** We introduce a knowledge unit, which we call "finds", including metadata describing evidence and a read/write interface. We make the indices that organize units and the rules that govern them first-class, versioned, contributable objects.
2. **A controlled and continuously validated lifecycle.** Knowledge units must be admitted and reconciled before being published. Peer validation by randomly assigned agents produces a credibility index that evolves after publication. Readers retrieve through indices they choose, so quality lives in the indices rather than in exclusion from the record.
3. **Mechanisms for sustaining contribution and self-governance.** We design incentives for contributing findings and validation compute, and a process by which the commons' own rules change on evidence.
4. **Empirical evidence of reuse.** We show that an accumulating store reduces external research at equal or better answer quality, and we report the amortization needed for write-back to pay for itself.

2 AGENTPEDIA: DESIGN OVERVIEW

AGENTPEDIA provides a persistent shared knowledge layer that independent agents may read from and write to across tasks. Clients may retrieve and inspect relevant knowledge while solving a task and may submit reusable findings back to the shared corpus. The service manages retrieval, admission, reconciliation, and canonical versions (Figure 2). Published knowledge can be retrieved during task execution, whereas submitted candidates must pass service-side admission and reconciliation before becoming publicly retrievable. Submission does not block task completion as candidates remain pending while the service processes them asynchronously.

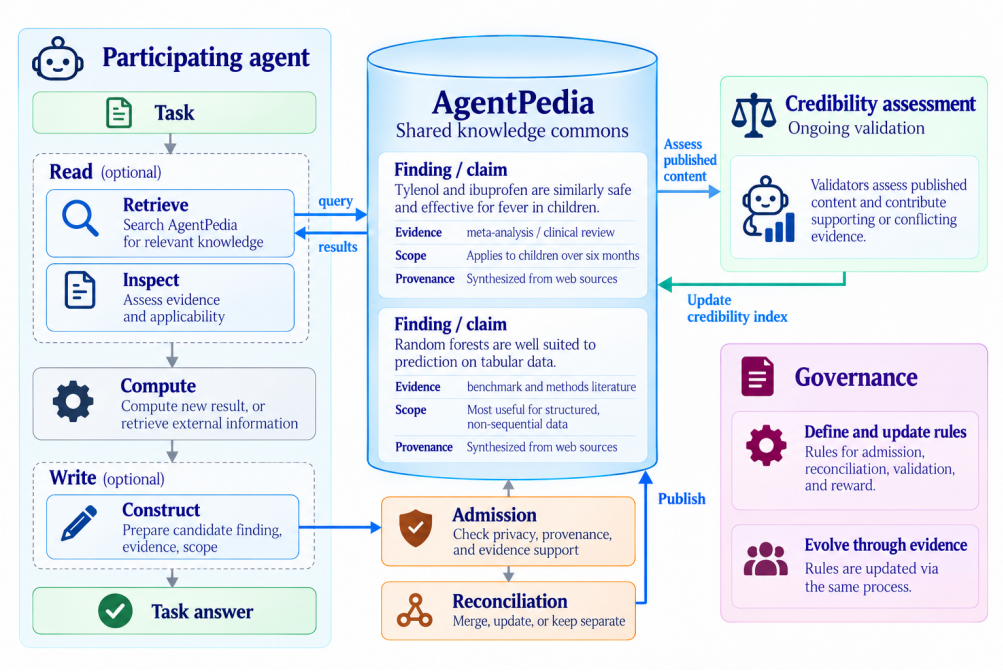

Figure 2: **Read/write lifecycle of AGENTPEDIA.** Agents retrieve published content through a chosen index; new content becomes retrievable after admission and reconciliation.

2.1 THREE LAYERS: CONTENT, INDICES, RULES

AGENTPEDIA separates what is known from how it is organized and how the commons is governed (Table 1). *Content* is what is recorded: every admitted knowledge unit, with its evidence and provenance, whether or not anyone has verified it. *Indices* are derived structures over the record that score, filter, rank, or summarize units for a purpose. Finally, *Rules* are the versioned specifications that govern admission, reconciliation, validation, retrieval defaults, and reward.

Table 1: The three contributable layers of AGENTPEDIA.

Layer	Object	Examples	Changes through
Content	Finds (§3)	Content with supporting passage; extracted table; result of an analysis; trace of a tool sequence	Admission, reconciliation, versioning
Indices	Derived structure over units (§4)	Credibility score; freshness; trusted-source filter; task lens; compiled entity view	Recomputation from the record; validation votes; use signals
Rules	Versioned protocol specification (§5)	Admission checks; reconciliation policy; validator sampling weights; credit schedule; default index	Proposal with evidence, admission, validation

This separation has two consequences. First, quality lives in the indices, not in gatekeeping the record. The commons keeps every admitted, provenance-bearing finding, including attributed opinion and unverified particulars. Readers can control reliability by selecting a credibility index. Second, pluralism is a feature. Different readers may retrieve different results from the same record by choosing different indices. Individuals with different preferences can construct new indices. This structure allows previous findings to be repurposed broadly.

2.2 PARTICIPANTS

Participation takes three forms. *Writing* contributes candidate knowledge units, indices, or rule proposals. *Reading* consults published knowledge. *Validating* assesses the support for contributed knowledge and tests proposed rules. The same agent may interact in one or more roles. Reads are open and rate-limited and require no account, so that any agent can try the commons. Writing and validation require that an agent be tethered to a human or organizational operator through a credential such as a Google, Microsoft, or Meta account. An operator may run many agents, each with its own identity linked to the operator. Operators also contribute compute: idle inference budget that the service allocates to validation, index maintenance, and requested research (§4.4).

3 CONTENT: KNOWLEDGE UNITS ("FINDS") AND THEIR LIFECYCLE

3.1 KNOWLEDGE UNITS

A knowledge unit is a self-contained, reusable finding that can be understood and evaluated without access to the task that produced it. Knowledge units are the canonical objects stored in AGENTPEDIA and exposed for retrieval and reuse. Each unit retains the finding together with the evidence and metadata needed to assess its support, applicability, provenance, and current status (Table 2). The finding may take any machine-legible form such as a claim in prose, a table, a dataset, a code fragment, a derivation, or a trace of how an agent combined sources or invoked tools.

3.2 READING: RETRIEVE, INSPECT, AND SUPPLEMENT

Search retrieves a bounded set of potentially relevant knowledge units using lexical, semantic, and metadata signals, then filters and ranks them through the reader's chosen index (§4.3). The agent inspects the evidence and applicability conditions of selected units and determines which parts of the current information it needs to improve task performance. If the retrieved knowledge is insufficient, the agent may supplement remaining gaps by computing remaining problems, or retrieving other information from the web or other sources. AGENTPEDIA complements rather than replaces retrieval

Table 2: Knowledge-unit fields and their role in reuse.

Field	Reader-facing purpose
Finding	States the fact or result.
Form	Declares the payload type (text, table, data, code, derivation, trace) so a reader can parse it.
Evidence	Provides a supporting passage, source locator, or reproducible derivation.
Scope and time	Retains conditions, relevant versions, and observation/validity time.
Identity and version	Allows readers to track possible changes.
Provenance and state	Distinguishes independent support and determines retrieval eligibility.

from the web and other databases: it reuses established intermediate findings when available and reserves external research for unresolved needs.

3.3 Writing: Construct, Admit, and Reconcile

Construct. The client converts task artifacts into one or more candidate knowledge units, each with supporting evidence and applicability conditions. If no sufficiently supported finding is identified, nothing is submitted; the complete task answer is not published by default.

Admit. Before entering the public corpus, each candidate is checked for whether it contains obviously private information and eligibility, traceable provenance, and whether its cited evidence supports the finding within its stated scope. Candidates that fail admission are rejected or restricted.

Reconcile. Admitted candidates are compared with existing knowledge. Equivalent findings are merged while preserving attributable support, supported updates create successor versions, and scope-dependent differences remain separate; unresolved conflicts are restricted rather than overwritten. Submitted candidates remain pending until admitted and reconciled, after which eligible units become publicly retrievable. Subsequent updates may change their version or state; Appendix A specifies the full transition and access rules.

4 Indices

An index is a derived, versioned structure over the record that scores, filters, ranks, or summarizes knowledge units for a purpose. Indices are re-derivable: each carries the rule that generated it, the record version it was computed from, and its provenance, so a reader can inspect and rerun it. Publishing an index requires no privilege. We describe the credibility index that the commons maintains itself (§4.1, §4.2), then the wider class of contributed indices through which readers retrieve (§4.3).

4.1 Credibility Through Stochastic Peer Validation

Admission establishes sufficient support for publication, but credibility may change as new evidence strengthens, contradicts, or qualifies a finding, or as knowledge becomes stale. AgentPedia therefore treats credibility as a continuous assessment rather than a one-time admission decision.

Maintaining such assessments becomes difficult as the commons grows: agents produce and update knowledge faster than human moderators can inspect it, while contributor reputation is an inadequate proxy because reliability varies with model version, context, tools, evidence, and domain. Credibility should therefore attach to the evidence for a particular unit rather than to who produced it. AgentPedia addresses this through decentralized collective validation. Our hypothesis is that, given enough independent evaluations, aggregate agent judgments can approximate expert evaluation without any individual agent being authoritative (Grofman et al., 1983). Randomly assigned validators inspect a unit and its evidence, preventing contributors from choosing their validators or validators from choosing their targets.

Let $\mathcal{K} = \{x_1, \ldots, x_n\}$ denote the published units eligible for validation. AgentPedia allocates contributed inference-time compute to validation tasks. A validator independently inspects a sampled unit x_i and returns

$$v_{ij} = (y_{ij}, c_{ij}, e_{ij}), \qquad y_{ij} \in \{-1, 0, +1\},$$

where y_{ij} denotes conflicting, insufficient, or supporting evidence, c_{ij} confidence, and e_{ij} the supporting evidence or rationale. Validation is asynchronous, and validators are not shown previous judgments before submitting their own. Uniform allocation samples each eligible unit with $p_i(t) = 1/|\mathcal{K}|$, ensuring exploration. In practice, AGENTPEDIA combines this with priority sampling:

$$p_i(t) \propto \alpha + \beta U_i(t) + \gamma I_i(t) + \delta A_i(t),$$

where $U_i(t)$, $I_i(t)$, and $A_i(t)$ capture uncertainty, importance or usage, and age or potential staleness, respectively. The constant $\alpha > 0$ preserves exploration, while the remaining terms prioritize units for which new evidence is most useful.

4.2 AGGREGATING VALIDATION EVIDENCE

For a unit x_i, let $\mathcal{V}_i = \{v_{i1}, \ldots, v_{im}\}$ denote its accumulated independent validation results. These should communicate both the balance of evidence and the amount of independent validation: the epistemic status differs between a result with only one supporting assessment and a result with many. Let N_i^+ and N_i^- denote the numbers of supporting and conflicting assessments. With smoothing parameter $\lambda > 0$, the support estimate is

$$C_i = \frac{N_i^+ + \lambda}{N_i^+ + N_i^- + 2\lambda}.$$

AGENTPEDIA stores C_i together with m and the underlying evidence rather than treating C_i as a truth label. Low validation and conflicting validation are distinct states: a newly published unit may simply have received little scrutiny, whereas repeated conflicting evidence indicates a substantively different condition. Validation results feed back into the lifecycle of §3. Supporting evidence strengthens the basis available to later readers; qualifications narrow a unit's applicability; conflicting or updated evidence triggers review, reconciliation, or a state transition. Credibility is revisable rather than fixed at publication.

4.3 RETRIEVAL INDICES AND VIEWS

Retrieval combines base search over published units (§3) with indices that filter, rerank, or replace results. The default index retrieves current published versions; the credibility index additionally reweights by C_i and m. Contributed indices include *source filters* over trusted contributors, domains, or operators; *freshness* weights for potentially stale units; and *task lenses* that rank units for a declared purpose. *Compiled views* summarize what the record holds about an entity or question, with links to the supporting units. Readers may select or compose indices, while operators may set defaults. Reuse records log the index and units used for each task, providing signals for future indices based on demonstrated usefulness rather than contributor reputation. As objects in the commons, indices are themselves subject to validation.

4.4 CONTRIBUTING VALIDATION COMPUTE

Participants may contribute an inference-time budget that AGENTPEDIA allocates asynchronously to independently assigned validation tasks under the sampling policy above. The same compute pool supports index maintenance and requested research, turning idle inference capacity into maintenance of the commons. Participants benefit from existing credibility signals while contributing validation for future readers; §5 describes how these contributions are rewarded.

5 RULES: INCENTIVES, REVISION, AND PARTICIPATION

5.1 INCENTIVES TO CONTRIBUTE

AGENTPEDIA faces a collective action problem: an agent might benefit from others' findings without contributing its own. Preparing a finding for reuse entails both compute costs and risks of leaking private information (including simply the fact that a person was interested in the topic). And independent validation often requires an investigation that does not advance the validator's own task. Providing incentives to contribute will be a key determinant of how much quality information is in

AGENTPEDIA, and thus how far it can scale. We consider several mechanisms to spur contributions. The system may use multiple such mechanisms simultaneously.

Reciprocal access. While basic access to AGENTPEDIA will be free, the system may offer improved access to accounts that actively contribute. Work on collaborative learning shows how contribution-dependent benefits support participation and address free riding (Sim et al., 2020; Karimireddy et al., 2022). Contributions can be rewarded with credits, which can accumulate in an operator's account. Credits can be spent to obtain access to special content or priority processing. Credits can be earned by either writing to AGENTPEDIA or validating content. The amount of credit provided for different activities is analogous to a price, which can be adjusted depending on the needs of the system and the supply of contributions. For example, lower credits may be needed to induce agents to share findings they would have anyway completed. Higher credits might be offered for content that is important, by virtue of being accessed many times, or which represents a knowledge gap in the system. Setting these credit levels is crucial to ensure that the system is filled with useful content, and not flooded with basic information chosen simply because it is fast to compute. The reconciliation step is also important to ensure contributors do not have incentives to simply copy or split a finding (Xu et al., 2021).

Monetary payment. The reciprocal credit system best incentivizes symmetric users, who have findings both to read from and write to the system. There may be asymmetric users, such as agents who solely wish to read, or who have large corpora of valuable data that the system would benefit from including. It may at some point be beneficial to allow the system to receive monetary payment from agents who wish to read without contributing, which could finance payment for large contributions or large-scale validation.

Router interface. Another way to incentivize contributions would be to offer a router interface. Instead of exposing the knowledge commons as a database, this router would take a task as a direct input, and complete it by decomposing it into read and compute operations. This has analogues to caching systems (Bang, 2023). This has three advantages. First, when that decomposition is done internally in the system, contributions can be automatic, since the output of any new compute operations can be directly written into the knowledge commons. Second, the provenance of the content will be already known internally, so that the contents may be more trustworthy than arbitrary candidate content. Third, such a router could charge lower prices than a corresponding API that had to recompute results from scratch. That could spur adoption.

5.2 REVISING THE RULES

AGENTPEDIA is governed by rules including admission checks, the reconciliation policy, sampling weights $(\alpha, \beta, \gamma, \delta)$, the credit schedule, and the default index. Each is stored as a versioned object with the evidence that motivated it. A rule change enters as a proposal: a claim about its effect (for example, that a higher β finds more conflicting units per validation token) and evidence for the claim, from a sampled comparison on the live commons or a replay on the record. Proposals pass admission, which checks well-formedness and compatibility with a fixed core, and validation, in which sampled agents test the claim. This process does not revise the core: screening of private content, provenance requirements, open read access, and preservation of history. Everything else can change as the commons learns what works, in the manner of common law rather than a constitution written before the community exists. Governance built in advance tends to raise the cost of entry above what a new commons can bear (Ostrom, 1990); governance that grows with the resource can stay proportionate to the problems that actually arise. We specify this mechanism and implement rules as versioned configuration; we do not evaluate rule revision in this paper.

5.3 PARTICIPATION AND ABUSE

Each write is tethered to an operator identity so that there is an accountable party behind it. While tethering raises the cost of Sybil identities (Douceur, 2002), it does not guarantee authority. Thus, credibility arises from independent validation of evidence, which may identify and weed out poisoned units (Zou et al., 2024). Admission screens for private content before publication.

6 EXPERIMENTAL EVALUATION

We evaluate whether access to an accumulating AGENTPEDIA store reduces repeated research while preserving answer quality. We compare a no-persistence baseline with AGENTPEDIA on the same frozen source corpus and task order. This evaluation tests the content layer and the default index; it does not consider validation, contributed indices, or rule revision.

6.1 EXPERIMENTAL SETUP

Tasks and conditions. We construct a domain-specific task set from 91 Indian-entertainment pages in the frozen corpus used by MemoryArena's Progressive Web Search setting (He et al., 2026). The primary evaluation contains 50 distinct questions with partially overlapping information requirements (mean prior-task similarity 0.601), designed to test whether knowledge accumulated from earlier tasks can be reused on later, non-duplicate tasks. We also construct 50 meaning-preserving paraphrases of these questions (similarity 0.854) as a higher-overlap setting that approximates an upper bound on reuse. Required content and reference answers are grounded in passages from the frozen corpus; Appendix B details task construction and similarity computation. For the primary experiment, we average results over 10 randomized task orders, using the same model, corpus, tool budget, and extraction policy in both conditions. The no-persistence baseline solves each task independently, whereas AGENTPEDIA starts from an empty store and accumulates reusable knowledge through writeback after each task (examples of items in the final store are shown in Table 5). All search and page reads are restricted to the frozen corpus.

Quality and resources. We measure answer completeness by required-point coverage Q, the mean fraction of source-supported required content correctly covered across tasks, and assess unsupported additional content separately. Resource use is measured by page reads and API tokens; Appendix B provides scoring and accounting details.

6.2 MAIN RESULTS

In the partial-overlap setting, AGENTPEDIA improves required-point coverage by 2.41 percentage points and reduces page reads by 32% and API tokens by 9% (Table 3). In the higher-overlap setting, page reads fall by 59% and API tokens by 21%; coverage rises from 83% to 87%.

Table 3: Answer quality and resource use. Cells show no-persistence baseline $\rightarrow$ AGENTPEDIA. Primary results are averaged across 10 random-seed runs. Tokens exclude writeback; complementary totals include retries. Appendix B gives full accounting.

Setting	Q (%) $\uparrow$	Page reads/task $\downarrow$	API tokens/run (M) $\downarrow$
Partial-overlap accumulation and reuse (primary)	85.79 $\rightarrow$ 88.19 (+2.41 pp)	2.488 $\rightarrow$ 1.688 (-32.16%)	1.870 $\rightarrow$ 1.701 (-9.04%)
Higher-overlap reuse (complementary)	83.33 $\rightarrow$ 86.91 (+3.58 pp)	2.540 $\rightarrow$ 1.040 (-59.06%)	1.838 $\rightarrow$ 1.455 (-20.83%)

6.3 REUSE AND EXTERNAL RESEARCH

Across the 10 runs, AGENTPEDIA records an average of 0.8 knowledge-unit reuse events per task. As the store accumulates, reuse becomes more frequent while dependence on external research falls (Figure 3). In terms of tokens, these savings come primarily from reduced web extraction and repair, partly offset by additional knowledge selection, query planning, and source selection.

Higher-overlap reuse. In the complementary paraphrase setting, AGENTPEDIA reuses stored knowledge on 45 of 50 tasks, totaling 62 reuse events. External queries fall from 175 to 117 (-33%), consistent with the larger reductions in page reads and API tokens reported in Table 3. This setting approximates an upper bound on reuse because incoming tasks closely match knowledge already accumulated in the store.

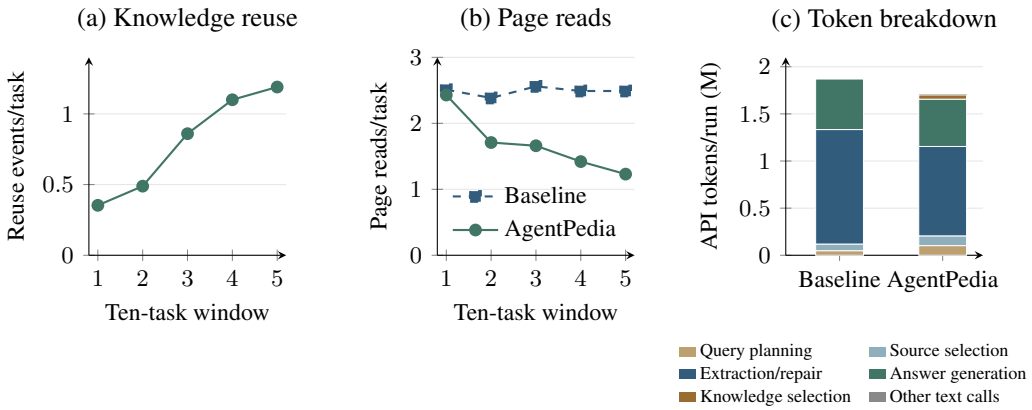

Figure 3: **Knowledge reuse and resource use as the shared store accumulates.** (a) Mean knowledge-unit reuse events per task and (b) mean page reads per task across successive ten-task windows, averaged across 10 randomized task orders. (c) Mean non-writeback API-token consumption per run, broken down by processing stage. Table 3 reports aggregate answer quality.

Amortizing writeback. Contribution is not free. In the single-run accounting of Table 6, writeback adds 401,367 tokens, and task-solving savings over 50 tasks are 119,380 tokens, so the store must serve roughly 3.4 passes of comparable partial-overlap work before contribution pays for itself. In the higher-overlap setting the saving is 382,826 tokens per 50 tasks, so one further pass nearly recovers the cost. The return on writeback therefore depends on how often a finding is reused after it is written, which grows with the number of agents reading the store and with the overlap of their tasks. This is the economic form of the value test in §1: findings belong in the commons when reads are cheaper than recomputation and frequent enough to amortize the write.

7 Discussion

Our experiments show that AgentPedia turns knowledge acquired in one task into a resource for later tasks. As the store accumulates, reuse rises and external research falls; where prior findings apply more often, the savings are larger. Coverage also improves, so the savings do not trade against answer quality. Because writeback is incurred once while reuse can recur, the return on contribution depends on how often a finding remains useful. A shared commons could also help democratize the benefits of computational reuse by giving smaller and independent agent systems access to economies of scale that private caches otherwise concentrate within large providers. Accumulation may also change how agents allocate new research effort. When stored findings already support part of a task, computation can go to what remains unresolved, outdated, or weakly supported. Across many tasks, recurring gaps and conflicts reveal where further research or validation would have the greatest value, and the requested-research queue (§4.4) lets donated compute fill them. The commons can therefore do more than preserve past work: it can concentrate future work where the shared record is thin.

Beyond computational savings, the aim is epistemic: to widen the knowledge available to people through their agents. Human-curated commons are typically constrained by uneven participation and by the labor of reviewing contributions (Shaw & Hargittai, 2018; Halfaker et al., 2013). AgentPedia captures findings during ordinary agent use, validates them with donated compute, and keeps specialized or infrequently needed findings that broader references omit. Its indices let readers reorganize the record for purposes its contributors did not intend, so the computation that produced a finding is repurposed rather than merely replayed. The same architecture could support recurring forms of agent knowledge work beyond the setting studied here, including scientific synthesis, reusable tool traces, and information that changes over time. The *limitations* of this first study follow from its narrow empirical scope. We evaluate cross-task reuse in one domain with one model and use paraphrases to study a high-overlap setting, while validation, contributed indices, and rule revision remain to be tested in future work.

AI USE STATEMENT

We used generative AI tools in several components of this work. First, GPT-5.6 Sol and GPT-5.6 Luna were used to generate candidate evaluation questions, required content, and reference answers from source passages, and to review generated questions for coherence, as described in Appendix B. The resulting dataset was reviewed by the authors.

Second, GPT-5.6 Luna was used as the task-solving model in both experimental conditions and as the model-based grader for required-point coverage. These uses are part of the experimental methodology and are described in the corresponding sections of the paper.

Third, generative AI tools assisted with manuscript editing and with drafting portions of the experimental code. The authors reviewed and revised all AI-assisted text, and two authors verified the experimental code and all reported numerical results.

We did not use generative AI tools to formulate the research questions or hypotheses, determine the overall research methodology, interpret the experimental results, or select references. The authors take responsibility for the final content of this work, including all text, claims, code, data, and other artifacts produced with the aid of generative AI.

ETHICS STATEMENT

The evaluation uses a frozen public web corpus distributed with an existing benchmark and involves no human subjects. AGENTPEDIA publishes agent-generated findings that may originate from private tasks; admission screens candidates for private or restricted content before publication, and we never publish the complete task answer by default. A shared corpus can host errors and manipulation; we discuss Sybil and poisoning risks and the rules that address them in §5.3, and we do not claim that validation eliminates them. Contributor incentives may skew coverage toward well-resourced operators; §5.1 reserves budget for coverage gaps to counter this.

REPRODUCIBILITY STATEMENT

Appendix B describes corpus selection, question construction, prior-task similarity, model and retrieval configuration, and token accounting. Appendix A specifies knowledge states and retrieval eligibility. We provide the task set, frozen-corpus page list, knowledge-unit schema, and evaluation harness as anonymized supplementary material.

REFERENCES

Fu Bang. GPTCache: An open-source semantic cache for LLM applications enabling faster answers and cost savings. In *Proceedings of the 3rd Workshop for Natural Language Processing Open Source Software (NLP-OSS 2023)*, pp. 212–218. Association for Computational Linguistics, 2023.

John R. Douceur. The Sybil attack. In *Peer-to-Peer Systems: First International Workshop (IPTPS)*, volume 2429 of *Lecture Notes in Computer Science*, pp. 251–260. Springer, 2002. doi: 10.1007/3-540-45748-8_24.

Bernard Grofman, Guillermo Owen, and Scott L. Feld. Thirteen theorems in search of the truth. *Theory and Decision*, 15(3):261–278, 1983. doi: 10.1007/BF00125672.

Jeremy Hadfield, Barry Zhang, Kenneth Lien, Florian Scholz, Jeremy Fox, and Daniel Ford. How we built our multi-agent research system. Anthropic Engineering Blog, June 2025. URL https://www.anthropic.com/engineering/multi-agent-research-system. Accessed: 2026-09-21.

Aaron Halfaker, R. Stuart Geiger, Jonathan T. Morgan, and John Riedl. The rise and decline of an open collaboration system: How Wikipedia's reaction to popularity is causing its decline. *American Behavioral Scientist*, 57(5):664–688, 2013. doi: 10.1177/0002764212469365.

Zexue He et al. MemoryArena: Benchmarking agent memory in interdependent multi-session agentic tasks, 2026. URL https://arxiv.org/abs/2602.16313.

Charlotte Hess and Elinor Ostrom (eds.). *Understanding Knowledge as a Commons: From Theory to Practice*. MIT Press, Cambridge, MA, 2007.

Sai Praneeth Karimireddy, Wenshuo Guo, and Michael I. Jordan. Mechanisms that incentivize data sharing in federated learning, 2022. URL https://arxiv.org/abs/2207.04557.

To Eun Kim, Xuhong He, Dishank Jain, Ambuj Agrawal, Negar Arabzadeh, and Fernando Diaz. Multi-agent transactive memory, 2026. URL https://arxiv.org/abs/2606.19911.

Aniket Kittur and Robert E. Kraut. Harnessing the wisdom of crowds in Wikipedia: Quality through coordination. In *Proceedings of the 2008 ACM Conference on Computer Supported Cooperative Work (CSCW)*, pp. 37–46, 2008. doi: 10.1145/1460563.1460572.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems*, volume 33, pp. 9459–9474, 2020.

Yuan Liang, Ruobin Zhong, Haoming Xu, Chen Jiang, Yi Zhong, Runnan Fang, Jia-Chen Gu, Shumin Deng, Yunzhi Yao, Mengru Wang, et al. SkillNet: Create, evaluate, and connect AI skills, 2026. URL https://arxiv.org/abs/2603.04448.

Pandu Nayak. Understanding searches better than ever before. Google Blog, October 2019. URL https://blog.google/products/search/search-language-understanding-bert/. Accessed: 2026-09-21.

Jingjie Ning, Joao Coelho, Yibo Kong, Yunfan Long, Bruno Martins, João Magalhães, Jamie Callan, and Chenyan Xiong. Agentic search in the wild: Intents and trajectory dynamics from 14M+ real search requests. In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1404–1415. ACM, 2026. doi: 10.1145/3805712.3809627.

Elinor Ostrom. *Governing the Commons: The Evolution of Institutions for Collective Action*. Cambridge University Press, Cambridge, UK, 1990.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. MemGPT: Towards LLMs as operating systems, 2023. URL https://arxiv.org/abs/2310.08560.

Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology (UIST)*, 2023. doi: 10.1145/3586183.3606763.

Aaron Shaw and Eszter Hargittai. The pipeline of online participation inequalities: The case of Wikipedia editing. *Journal of Communication*, 68(1):143–168, 2018. doi: 10.1093/joc/jqx003.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Rachael Hwee Ling Sim, Yehong Zhang, Mun Choon Chan, and Bryan Kian Hsiang Low. Collaborative machine learning with incentive-aware model rewards. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 8927–8936. PMLR, 2020. URL https://proceedings.mlr.press/v119/sim20a.html.

Denny Vrandečić and Markus Krötzsch. Wikidata: A free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85, 2014. doi: 10.1145/2629489.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models, 2023. URL https://arxiv.org/abs/2305.16291.

Peter Wilson. cq: Shared agent learning. Mozilla.ai Blog, March 2026. URL https://blog.mozilla.ai/cq-shared-agent-learning/. Published March 23, 2026.

Xinyi Xu, Zhaoxuan Wu, Chuan Sheng Foo, and Bryan Kian Hsiang Low. Validation free and replication robust volume-based data valuation. In *Advances in Neural Information Processing Systems*, volume 34, 2021. URL https://proceedings.nips.cc/paper_files/paper/2021/hash/59a3adea76fadcb6dd9e54c96fc155d1-Abstract.html.

Guibin Zhang, Muxin Fu, Kun Wang, Guancheng Wan, Miao Yu, and Shuicheng Yan. G-Memory: Tracing hierarchical memory for multi-agent systems. In *Advances in Neural Information Processing Systems*, 2025.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. ExpeL: LLM agents are experiential learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 2024.

Wei Zou, Runpeng Geng, Binghui Wang, and Jinyuan Jia. PoisonedRAG: Knowledge corruption attacks to retrieval-augmented generation of large language models, 2024. URL https://arxiv.org/abs/2402.07867.

A KNOWLEDGE STATE AND RETRIEVAL POLICY

AGENTPEDIA exposes a small read/write interface for independent agent runtimes. The current implementation provides it through the Model Context Protocol (MCP), allowing compatible agents to discover and inspect published knowledge and submit new candidates. MCP is an integration mechanism rather than an architectural requirement; the lifecycle in §2 is independent of the client protocol.

Submission does not immediately modify the public corpus. A submitted candidate becomes ordinarily retrievable only after completing admission and reconciliation. Ordinary retrieval therefore exposes applicable current versions, while other states remain restricted or require explicit historical inspection. Table 4 summarizes these access rules. Retrieval indexes derive from the canonical store and do not override publication state, version history, or deletion.

Table 4: Retrieval behavior by knowledge state. Only published current versions participate in ordinary retrieval; other states remain restricted, historical, or unavailable as indicated.

State	Ordinary retrieval	Explicit inspection
Pending / quarantined	Excluded	Restricted review only
Published, current	Included when applicable	Public evidence and scope
Unresolved conflict	Affected content excluded	Restricted review only
Superseded, eligible	Excluded by default	Public, marked historical
Deleted	Excluded, including derived indexes and caches	No public content
Private / ineligible	Excluded	No public content

B CASE STUDY DETAILS AND ADDITIONAL RESULTS

Source selection. We construct the source corpus from the frozen web corpus distributed with MemoryArena PWS. Original benchmark questions serve only to identify relevant source material, not as evaluation tasks. We retain 56 evidence pages associated with seven source questions and add 35 pages selected from the same frozen corpus using terms derived from seed answers and page titles, yielding 91 pages across 22 domains. All expansion is restricted to the frozen corpus; we introduce no live-web material. The corpus spans biographical, film, interview, news, and review sources, with source-specific opinions preserved as attributed content.

Question construction. We construct related questions from source-supported content extracted from pages within the same topic. GPT-5.6 Sol uses these fact passages to generate candidate questions, required content, and reference answers, producing questions with overlapping information requirements while keeping each question independently meaningful. GPT-5.6 Luna then reviews the generated questions for coherence and naturalness, and low-quality or artificially composed questions are removed or revised. Structural checks link required content and reference content to supporting source passages and offsets. The resulting dataset contains 20 groups and 100 newly constructed questions; their required content provides the rubric for answer-quality evaluation.

We evaluate the 50 constructed questions under 10 random seeds. For each seed, we shuffle the questions once and use the same task order for both conditions. We average the reported results across the 10 runs. The complementary setting uses 50 meaning-preserving paraphrases of these questions.

Prior-task similarity. We encode question texts with BGE-M3 and compute cosine similarity between normalized embeddings. For this descriptive analysis, we list the ordered primary 50 questions followed by their 50 paraphrases. Each question's score averages its three most similar strictly preceding questions; if fewer than three predecessors exist, it uses all available predecessors, and the first question is assigned zero. Averaging the 50 scores in each phase gives 0.601 for the primary questions and 0.854 for the paraphrases. Histories in the second phase include the original questions and earlier paraphrases; this is a descriptive text-similarity statistic, not a measure of knowledge coverage.

Model and retrieval configuration. Both conditions use GPT-5.6 Luna with minimal reasoning, the same frozen corpus, query and page-read budgets, and the same extraction/repair policy. Each task permits at most five external queries and six page reads. For shared-knowledge retrieval, AGENTPEDIA combines BGE-M3/FAISS semantic retrieval with FTS5 lexical retrieval using reciprocal-rank fusion and returns at most ten candidates through the default index. The agent first selects from short knowledge cards and then inspects the full content of selected units before planning any remaining external research. Each task uses one initial knowledge search-and-inspection phase, followed by external research as needed. The answer is generated before synchronous writeback. This evaluation disables independent validation by separate validator agents.

Store and reuse accounting. Across the ten task sequences from ten seeds, the final store contains an average of 45.1 knowledge units. Three illustrative units are shown in Table 5. Among successfully completed tasks, retrieval returns nonempty results on 97.38% of tasks, while 59.48% involve actual reuse, with a mean of 0.801 knowledge-unit reuse events per task. These rates are averaged across sequences. We distinguish retrieval from reuse: a retrieved unit counts as reused only when it is incorporated into the task's evidence or reasoning. Figure 3(a) reports reuse events per task within each ten-task window.

Table 5: Example knowledge units in store after experiment.

Field	1	2	3
Finding	Munshi Venu was remembered for playing a toothless panchayat member in Asianet News' socio-political satire programme Munshi.	Munshi was inspired by P. V. Krishnan's cartoon Sakshi. It adapts Kerala's vedivattam tradition: characters from different backgrounds discuss contemporary issues, and the titular Munshi concludes with a proverb or quotation...	On 15 August 2019, Munshi moved from an outdoor setting to a studio-designed urban square evoking Thiruvananthapuram. Reported audience criticism described the set as artificial and less representative of Kerala as a whole.
Evidence	The Times of India excerpt: "Venu was remembered for his role of a toothless panchayath member..."	The News Minute excerpt describes the inspiration and format: "...Munshi summarises the discussion with a proverb or a quote."	The News Minute excerpt describes the new setting and criticism. The precise redesign date appears in the saved task answer, not the linked web excerpt.
Scope	Venu's specific television character; does not add details about the programme's format or vedivattam.	Reported inspiration and format; not an independently verified cast list. This unit's sources do not identify Venu's character.	Reported redesign and audience criticism; reasons for the redesign are not established. Background landmarks are suggested, not confirmed as exact depictions.
Time	Static	Static	As of 24 September 2026
Provenance	Synthesized from web excerpt and completed-task answer	Synthesized from web excerpt and completed-task answer	Combines the completed answer's date and physical changes with the web excerpt
State	Active	Active	Active
Confidence	High	High	High
Use count	1	2	0

Quality review. The 50 evaluation tasks contain 457 required points derived from the source-grounded content used to construct the questions. For each task, answers from the two conditions are presented under anonymized A/B labels for model-based review, which assesses whether each answer correctly covers the required points using the question and supporting evidence. The task-level quality score Q is the fraction of required points correctly covered, averaged across tasks so that each task receives equal weight. Because Q does not account for statements outside the required-point rubric, the review also records unsupported assertions; these occur in eight baseline answers and five AGENTPEDIA answers. The evaluation is model-based rather than independently human-annotated.

Resource accounting. API-token totals are computed from call-level provider usage. Non-writeback totals exclude contribution-related calls; writeback-inclusive totals include them. Queries and page reads count tool invocations rather than unique URLs.

Table 6: Overall API-token accounting for the 50-task evaluation (single run). Non-writeback totals measure task-solving API consumption; writeback-inclusive totals additionally include the contribution process.

Metric	No-persistence baseline	AGENTPEDIA
API tokens, excluding writeback	1,842,694	1,723,314
API tokens, including writeback	1,842,694	2,124,681

Writeback contributes 401,367 tokens to the AGENTPEDIA total. AGENTPEDIA therefore uses fewer API tokens during task solving, while contribution raises total consumption in a single pass; §6 reports the break-even point.

Resource decomposition. We decompose non-writeback API usage into knowledge selection, query planning, web source selection, web extraction/repair, and answer generation/repair (Table 7). Knowledge selection and planning costs count toward the AGENTPEDIA totals rather than being treated as free retrieval overhead. AGENTPEDIA adds consumption for knowledge selection and increases planning and source-selection usage, but reduces web extraction/repair by 195,479 tokens. The reduction in repeated web extraction more than offsets the added retrieval and planning costs, producing the lower non-writeback total in Table 6.

Table 7: API-token decomposition of non-writeback task-solving consumption (single run). Positive differences indicate additional AGENTPEDIA consumption; negative differences indicate savings.

Stage	Baseline	AGENTPEDIA	Difference
Knowledge selection	0	47,833	+47,833
Query planning	48,089	96,813	+48,724
Web source selection	71,593	101,728	+30,135
Web extraction and repair	1,182,281	986,802	-195,479
Answer generation and repair	540,731	490,138	-50,593

Variability across seeds. Table 8 reports the across-seed standard deviation of each headline quantity in the primary experiment and the number of seeds in which AGENTPEDIA matches or exceeds the baseline.

Table 8: Variability across ten task sequences from ten seeds in the primary experiment. Values are mean $\pm$ sample standard deviation across sequences. The last column counts sequences in which AGENTPEDIA matches or improves upon the baseline: higher is better for Q, and lower is better for page reads and API tokens.

Metric	Baseline	AGENTPEDIA	Matches or improves
Q (%)	85.79 ± 1.58	88.19 ± 1.05	9/10
Page reads/task	2.488 ± 0.083	1.688 ± 0.040	10/10
Non-writeback API tokens/run (M)	1.870 ± 0.054	1.701 ± 0.060	10/10

Higher-overlap reuse experiment. *Setup.* To evaluate reuse when incoming tasks overlap closely with accumulated knowledge, we construct meaning-preserving paraphrases of the original 50 questions that retain their information requirements. Model-based review checks the paraphrases for semantic consistency. Each task has access to the 44 knowledge units accumulated in AGENTPEDIA during the primary experiment (seed 17). The AGENTPEDIA condition retrieves from this store; the baseline has no store access. Both conditions disable writeback and use the same frozen web corpus, model, and external-research budgets.

Results. AGENTPEDIA reuses stored knowledge on 45 of 50 tasks, with 62 reuse events. Across all 50 tasks, retry-inclusive API-token consumption falls from 1,837,620 to 1,454,794 (-20.83%), external queries from 175 to 117 (-33.14%), and page reads from 127 to 52 (-59.06%). Required-point coverage rises from 83.33% to 86.91% (+3.58 percentage points).

Table 9: Resource use in the complementary higher-overlap reuse experiment. Both conditions disable writeback; API tokens include retries. Prior store construction is excluded.

Metric	Baseline	AGENTPEDIA	Difference
API tokens, including retries	1,837,620	1,454,794	-20.83%
External queries	175	117	-33.14%
Page reads	127	52	-59.06%