Journal of Science, Innovation & Social Impact

2026 2nd International Forum on Computing and Intelligent Systems (IFCIS 2026)

Article

Comparative Empirical Evaluation of Prompting Strategies for Large-Language-Model Web-Navigation Agents on the WebArena Benchmark

Xinyu Zhang[1,*], Yixin Zheng[2] and Chenhui Hao[3]

[1] Applied Computing, University of Toronto, Toronto, ON, Canada
[2] Information Science, University of Michigan, Ann Arbor, MI, USA
[3] Civil Engineering, University of California, CA, USA
* Correspondence: Xinyu Zhang, Applied Computing, University of Toronto, Toronto, ON, Canada

Abstract: Large language models (LLMs) are increasingly deployed as autonomous agents that operate web interfaces on behalf of users, yet reported success rates vary widely across studies that differ simultaneously in backbone model, prompting strategy, and evaluation environment, making it difficult to attribute performance to any single factor. This paper presents a controlled factorial comparison that isolates two of these factors on a fixed environment. We evaluate three backbone LLMs (GPT-4, Llama-3-70B-Instruct, and GPT-3.5) crossed with four prompting strategies (direct prompting, ReAct-style interleaved reasoning, Reflexion-style episodic retry, and best-first lookahead) on all 812 tasks of the WebArena benchmark, with a 165-instance WorkArena-L1 robustness check, under an identical action space, observation format, and step budget. Backbone choice accounts for the dominant share of variation: the weakest GPT-4 configuration (15.2%) exceeds the strongest GPT-3.5 configuration (9.7%) by 5.5 percentage points. Strategy gains are moderate and compound with backbone strength, ranging from 3.6 points on GPT-3.5 to 8.4 points on GPT-4. A cost analysis shows that the marginal cost of one additional percentage point rises roughly tenfold along the GPT-4 strategy ladder. All configurations remain far below the 78.24% human reference, indicating substantial remaining headroom.

Keywords: web-navigation agents; large language models; prompting strategies; empirical evaluation

Received: 29 April 2026
Revised: 17 June 2026
Accepted: 30 June 2026
Published: 03 July 2026

1. Introduction

1.1. Background and Motivation

Autonomous web-navigation agents built on large language models have progressed from controlled demonstrations to early commercial deployment within roughly three years. The WebArena benchmark established a reproducible reference point for this progress by hosting six fully functional websites and verifying task completion programmatically, reporting a 14.41% success rate for a GPT-4 baseline against a 78.24% human reference [1]. Complementary resources extended the empirical picture along different axes: Mind2Web collected 2,350 demonstrations over 137 real websites to study generalization across tasks, sites, and domains [2]. WebVoyager showed that multimodal agents can complete 59.1% of live-web tasks when evaluated by a model-based judge [3,4]. Despite this rapid accumulation of benchmarks and techniques, published numbers are difficult to compare. Studies differ simultaneously in backbone model, prompting strategy, observation encoding, action space, and step budget, so a reported improvement may reflect any combination of these choices [5,6]. Leaderboard entries compound the

problem: submissions frequently bundle a proprietary backbone, a custom observation pipeline, and an undisclosed amount of inference-time computation into a single headline number, leaving the individual contribution of each ingredient unrecoverable [7]. A practitioner deciding whether to invest in a stronger backbone or in a more elaborate prompting scaffold currently has little controlled evidence to consult, since few studies vary one factor while holding the others fixed, and fewer still report the inference cost at which their gains were purchased.

1.2. Research Questions and Contributions

This study addresses the attribution gap with a deliberately narrow design: a full factorial comparison of three backbones and four prompting strategies on one fixed environment, with every other implementation detail held constant.

1.2.1. Research Questions

Three questions structure the analysis. RQ1 asks how much of the observed performance variation on WebArena is attributable to backbone choice as opposed to prompting strategy when both are varied under identical conditions. RQ2 asks whether strategy-induced gains are uniform across the six WebArena site categories or concentrated in particular interaction patterns, and whether they transfer to an enterprise task distribution. RQ3 asks what each strategy costs in inference spending per percentage point of success gained, a quantity that published leaderboards rarely disclose yet practitioners must budget for. The questions are ordered by decision priority: a deployer first selects a backbone, then a scaffold, and finally validates whether the chosen pair justifies its operating cost [8,9]. The third question gains urgency from evidence that agent performance on realistic, time-consuming tasks collapses well below headline benchmark numbers, leaving budget-constrained deployment decisions especially underinformed [10].

1.2.2. Contributions

The paper makes three contributions. The first is a 3×4 factorial evaluation covering all 812 WebArena tasks for every cell, with three seeds per cell, which to our knowledge is the largest controlled strategy comparison on this benchmark. The second is a per-site and cross-benchmark breakdown that locates where strategy gains concentrate, supported by a 165-instance WorkArena-L1 replication. The third is a failure-mode and cost analysis that quantifies the diminishing economic efficiency of inference-time scaffolding, yielding concrete guidance on when additional prompting machinery stops paying for itself. The study deliberately proposes no new agent architecture; its value lies in measurement discipline applied to existing, widely adopted techniques under conditions that make their effects separable.

2. Related Work

2.1. Web-Agent Benchmarks

2.1.1. Simulated and Sandboxed Benchmarks

Early work framed web interaction as reinforcement learning over synthetic miniature interfaces, with workflow-guided exploration over scripted MiniWoB widgets establishing the paradigm [11]. WebShop scaled the simulated setting to an e-commerce site populated with 1.18 million real products and 12,087 crowd-sourced instructions, exposing a 30-point gap between the best agent and human experts [12,13]. WebArena moved the field toward functional realism by self-hosting complete open-source applications and checking end states programmatically, a design this study adopts as its primary environment [14]. VisualWebArena added 910 visually grounded tasks on which the strongest vision-language configuration reached 16.4% against an 88.7% human reference, confirming that realism rather than modality is the dominant difficulty driver [15,16].

2.1.2. Live-Web and Enterprise Benchmarks

A second line of work evaluates agents on live or enterprise-grade interfaces [17,18]. WorkArena instantiated 33 atomic knowledge-work task templates, expanding to 19,912 unique instances, on a production ServiceNow platform [19]. Its compositional extension chains these atoms into 682 multi-step workflows that remain largely unsolved [20,21]. Live-web suites trade reproducibility for ecological validity: page content shifts between runs, which complicates seed-controlled comparison. This study uses the sandboxed WebArena for its main factorial design precisely because functional verification and frozen content allow strategy effects to be isolated, and uses WorkArena-L1 as a controlled enterprise replication rather than an open-web one, accepting the external-validity cost that this choice entails.

2.2. Prompting Strategies for LLM Agents

Inference-time scaffolding for LLM agents has developed along several directions. Interleaved reasoning, episodic self-critique, and inference-time search form one family, and the three representatives evaluated in this study are specified in Section 3.2. A second family augments the prompt with retrieved experience: Synapse selects trajectory-level exemplars from a memory of past episodes and approached saturation on MiniWoB++ while improving step-level accuracy on Mind2Web [22,23]. A third family changes the observation modality rather than the reasoning pattern: SeeAct showed that GPT-4V completes 51.1% of live Mind2Web tasks when paired with oracle element grounding, although performance drops substantially under realistic grounding [24]. These strategies were introduced on different backbones and benchmarks, with heterogeneous step budgets and observation encodings [25,26]. This heterogeneity is precisely the comparability gap the present factorial design addresses [27]. None of the published comparisons crosses strategy families with backbone tiers on a single environment, so the interaction between the two factors, central to the budgeting question raised in Section 1, has remained unmeasured [28].

3. Evaluation Methodology

Methodological studies caution against naive cross-paper comparison. AgentBench evaluated 29 models across eight interactive environments and found that most failures fall into a small taxonomy of recurring error types, with weaker models disproportionately producing malformed actions [29]. Its multi-environment evidence also showed that model rankings shift across environment families, implying that strategy comparisons are only interpretable when the environment, observation format, and action space are pinned. Reported gaps between commercial and open-weights backbones were larger in that study than gaps between adjacent prompting variants, a pattern the present results corroborate under tighter controls [30,31]. This study follows the resulting prescription: a single primary environment, one shared action grammar, programmatic rather than model-based verification, and matched-task statistical testing.

4. Experimental Setup

4.1. Benchmarks and Environment Configuration

The primary environment is WebArena, comprising 812 tasks instantiated from 241 templates across six self-hosted sites: an e-commerce storefront (Shopping), its administration console (Shopping Admin), a GitLab instance, a Reddit-style forum, an OpenStreetMap deployment, and a set of tasks spanning multiple sites [32]. Success is verified by executable checks on the final environment state, removing judge bias from the measurement. The secondary environment is WorkArena-L1, from which we sample 5 instances per template for all 33 templates, yielding 165 enterprise-flavored instances on a ServiceNow developer instance. Both environments are accessed through the BrowserGym ecosystem, which supplies a unified accessibility-tree observation and a shared primitive action vocabulary of clicks, typing, scrolling, tab control, and navigation, and whose cross-benchmark experiments document that backbone rankings can change

across benchmark families [33,34]. Episodes terminate on an explicit stop action or after 30 steps, a budget under which the original benchmark study observed diminishing returns for further steps. Observations are truncated to 4,096 tokens per turn for every configuration so that context-window differences between backbones do not leak into the comparison; truncation keeps the most recent page content and discards the oldest interaction history first. All WebArena containers are reset to identical snapshots before each seed, and the WorkArena instance is restored from a fixed state between episodes, so no configuration benefits from residue left by another, and write actions performed by one episode cannot alter the verification targets of the next. Table 1 summarizes both environments; the human reference and original GPT-4 baseline are reproduced from the benchmark publication for orientation.

Table 1. Evaluation Environments. Human Reference and Original Baseline Are Reported by the Respective Benchmark Papers; All Remaining Counts Are Taken from Official Documentation.

Property	WebArena	WorkArena-L1 (sampled)
Tasks evaluated	812	165
Templates	241	33
Sites / platform	6 self-hosted sites	ServiceNow developer instance
Verification	Programmatic state check	Programmatic state check
Human reference	78.24%	not reported
Original GPT-4 baseline	14.41%	—
Max steps per episode	30	30

4.2. Agents under Test

4.2.1. Backbone Language Models

Three backbones span roughly two orders of magnitude in inference price and include one open-weights representative. GPT-4 (gpt-4-0125-preview) represents the frontier closed tier; GPT-3.5 (gpt-3.5-turbo-0125) represents the economical closed tier; Llama-3-70B-Instruct, served locally through vLLM on four A100 GPUs, represents reproducible open weights. Open-weights backbones trained jointly on natural language and code have been shown to narrow the agent-capability gap to closed models, making Llama-3-70B a meaningful midpoint rather than a strawman [35]. Decoding uses temperature 0.2 and nucleus threshold 0.95 for all backbones; the same system prompt, exemplars, and action grammar are shared verbatim across backbones so that only the underlying model varies. Checkpoint identifiers are pinned for the closed models and the open-weights model is loaded from a fixed snapshot, so the grid reflects a single point in time for each provider rather than a moving target.

4.2.2. Prompting Strategies

Four strategies form an ascending ladder of inference-time computation. Direct prompting emits one action per observation with no intermediate text. The second strategy interleaves a short reasoning passage before each action, following the ReAct recipe that improved WebShop success by ten absolute points with only one or two in-context exemplars [36]. The third grants each task one retry: after a failed first episode, a verbal self-critique is generated and prepended to the second attempt, adapting the Reflexion mechanism that achieved large gains on coding and decision-making tasks without any weight updates [37]. The fourth, lookahead, performs best-first search over candidate actions with branching factor 3 and depth 2, scoring branches with the same backbone acting as value estimator, a configuration adapted from inference-time tree search for language-model agents, which reported a 28.0% relative improvement on WebArena at substantially increased token cost [38]. Branch expansion is simulated on cloned environment state so that rejected branches leave no side effects, and the

committed action is re-executed on the live state. Hyperparameters for each strategy were fixed once on a 40-task development subset before the full runs and then frozen; no per-site or per-backbone tuning occurred afterward, and identical settings apply to every cell of the grid. Table 2 specifies each strategy; per-step LLM calls translate directly into the cost figures analyzed in Section 4.3.

Table 2. Strategy Configurations. All Strategies Share the Action Grammar, Observation Format, Exemplars, and 30-Step Budget; They Differ Only in the Columns Shown.

Strategy	Reasoning trace	Retry on failure	Search width × depth	LLM calls per step
Direct	none	no	—	1
ReAct	per-step	no	—	1
Reflexion	per-step	1 retry	—	1 (×2 episodes)
Lookahead	per-step	no	3 × 2	4–7

4.3. Evaluation Protocol

4.3.1. Metrics and Functional Verification

The primary metric is task success rate (SR), the fraction of tasks whose programmatic end-state check passes. Secondary metrics are mean trajectory length in environment steps and mean inference cost per task in USD, computed from provider prices for closed models and from amortized GPU-hour cost for the local backbone at the time of experimentation. Verification design matters for measurement validity: expert annotation of 1,302 web-agent trajectories has shown that rule-based evaluators systematically underestimate capability while model-based judges overestimate it [39]. Relying exclusively on WebArena's executable checks keeps the bias direction known and uniform across cells, at the price of occasional false negatives on underspecified tasks, a limitation revisited in Section 5. Cost accounting attributes every token billed or served during an episode to that episode, including the value-estimation calls issued by the search strategy and the discarded first attempt of the retry strategy, so the reported per-task figures reflect what a deployer would actually pay rather than the cost of the successful trajectory alone. Trajectory lengths are recorded over all episodes, successful or not, to avoid survivorship distortion.

4.3.2. Statistical Validation

Every cell of the 3×4 design is run with three random seeds covering all 812 tasks, and reported values are means with standard deviations across seeds. Seeds control sampling randomness in decoding and the instantiation order of episodes; environment content is identical across seeds by construction. Within a backbone, strategies are compared with McNemar's test on matched task identifiers, pooling seeds by majority outcome; differences are called moderate only when the two-sided p-value falls below 0.01, and all pairwise strategy contrasts on GPT-4 meet this bar while the ReAct-to-retry contrast on the two weaker backbones does not. Wilson 95% confidence intervals on a single 812-task run span roughly ±2.3 points at the observed success levels, which is wider than most strategy gaps on the weaker backbones, underscoring why matched-pair testing rather than interval overlap is used for inference.

5. Results and Analysis

5.1. Aggregate Success Rate

Table 3 reports the full factorial grid, and Figure 1 visualizes it. The best configuration, GPT-4 with lookahead, solves 23.6% of tasks; the weakest, GPT-3.5 with direct prompting, solves 6.1%. Our GPT-4 direct cell (15.2%) sits 0.8 points above the originally published 14.41% baseline, consistent with the intervening model revision from the 0613 to the 0125-preview checkpoint under an otherwise comparable text-only setup.

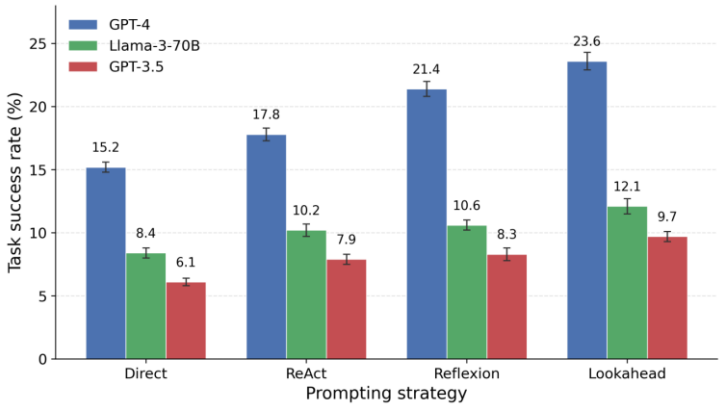

Figure 1. WebArena Success Rate Across Twelve Backbone-Strategy Configurations

Table 3. WebArena Task Success Rate (%), Mean ± Standard Deviation over 3 Seeds of 812 Tasks Each. Bold Marks the Strongest Backbone in Each Strategy Column.

Backbone	Direct	ReAct	Reflexion	Lookahead
GPT-4	15.2 ±0.4	17.8 ±0.5	21.4 ±0.6	23.6 ±0.7
Llama-3-70B	8.4 ±0.4	10.2 ±0.5	10.6 ±0.4	12.1 ±0.6
GPT-3.5	6.1 ±0.3	7.9 ±0.4	8.3 ±0.5	9.7 ±0.4

Task success rates of three backbones under four prompting strategies, with error bars denoting standard deviation over three seeds. GPT-4 dominates at every strategy level, rising from 15.2% under direct prompting to 23.6% under lookahead. The weakest GPT-4 configuration exceeds the strongest GPT-3.5 configuration (9.7%) by 5.5 percentage points and the strongest Llama-3-70B configuration (12.1%) by 3.1 points. Within-backbone spreads widen with backbone strength, from 3.6 points on GPT-3.5 to 8.4 points on GPT-4.

5.1.1. Backbone Effects

Backbone choice is the dominant factor. Averaging across strategies, GPT-4 reaches 19.5%, Llama-3-70B 10.3%, and GPT-3.5 8.0%, so the mean gap between the strongest and weakest backbone (11.5 points) exceeds the largest within-backbone strategy gap (8.4 points). The ordering is strict in every column of Table 3, and no prompting strategy lifts a weaker backbone past a stronger one under any configuration tested. Published WebArena results bracket our grid in an informative way. A fine-tuned open agent built on a 6B base reports 18.2%, between our Llama-3-70B and GPT-4 bands [40]. A stacked-policy method reports 33.5% with GPT-4, ten points above our best cell, indicating that task-specific policy decomposition can extract more from the same backbone than the generic strategies studied here [41].

5.1.2. Strategy Effects

Strategy gains are consistent in direction yet moderate in size, and they compound with backbone strength. Moving from direct prompting to lookahead adds 8.4 points on GPT-4, 3.7 on Llama-3-70B, and 3.6 on GPT-3.5. The retry strategy shows the sharpest interaction: its margin over per-step reasoning alone is 3.6 points on GPT-4 yet only 0.4 on both weaker backbones, suggesting that profiting from verbal self-critique requires the very capability the weaker backbones lack, namely reliably diagnosing why an episode failed. This pattern matches evidence that search- and critique-style methods help only when the discriminating model is sufficiently accurate, with gains vanishing below a discrimination-accuracy threshold [42]. Lookahead yields the largest single-strategy gain on every backbone, and our GPT-4 lookahead cell (23.6%) is directionally consistent with the 19.2% reported for tree search over a GPT-4o policy under a different search budget, with the residual difference plausibly attributable to the larger branching configuration and the checkpoint gap between the two studies.

5.2. Per-Site Difficulty and Cross-Benchmark Robustness

Table 4 decomposes GPT-4 performance across the six WebArena site categories, and Figure 2 visualizes the same grid. The site ordering is stable across all four strategies: Map is easiest (21.1% to 31.2%), followed closely by Reddit and Shopping, while Shopping Admin (11.0% to 18.1%) and GitLab (11.1% to 19.4%) lag, and multi-site tasks remain hardest (6.3% to 12.5%). Strategy gains are largest in absolute terms on the configuration-heavy sites, with GitLab improving 8.3 points from direct prompting to lookahead, and smallest on multi-site tasks, where cross-application state tracking appears to bottleneck before action selection does. The pattern is consistent with the mechanics of the scaffolds: deeper deliberation pays where individual pages are dense with similar controls, yet adds little when the missing ingredient is memory carried across applications.

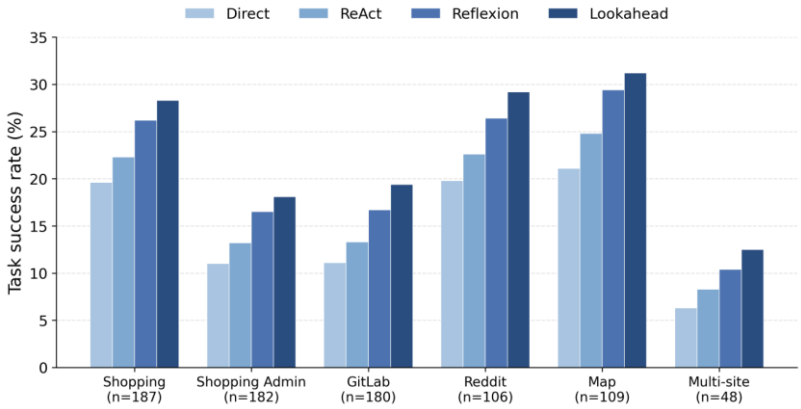

Figure 2. Per-Site Success Rates of GPT-4 under Four Prompting Strategies

Table 4. Per-site Success Rate (%) of GPT-4 on WebArena, Mean over 3 Seeds. Site Task Counts Shown in Parentheses; Count-Weighted Row Means Reproduce the GPT-4 Row of Table 3 within Rounding.

Strategy	Shopping (187)	Shopping Admin (182)	GitLab (180)	Reddit (106)	Map (109)	Multi-site (48)
Direct	19.6	11.0	11.1	19.8	21.1	6.3
ReAct	22.3	13.2	13.3	22.6	24.8	8.3
Reflexion	26.2	16.5	16.7	26.4	29.4	10.4
Lookahead	28.3	18.1	19.4	29.2	31.2	12.5

Success rates of GPT-4 across the six WebArena site categories. The site ordering is invariant to strategy: Map peaks at 31.2% under lookahead while multi-site tasks remain lowest at 12.5%. The largest strategy-induced improvement occurs on GitLab (+8.3 points from direct prompting to lookahead) and the smallest on multi-site tasks (+6.2 points), indicating that inference-time scaffolding helps most where single-application navigation depth, rather than cross-application state tracking, is the binding constraint.

The WorkArena-L1 replication in Table 5 confirms that the two headline findings transfer to an enterprise distribution. Backbone ordering is preserved, lookahead improves every backbone, and the GPT-4 gain (+9.7 points, from 40.0% to 49.7%) again exceeds the gains on Llama-3-70B (+6.1) and GPT-3.5 (+4.8). Absolute levels are higher than on WebArena because atomic enterprise tasks are shorter and more form-like, yet the relative structure of the grid is unchanged [43,44]. The agreement across two environments with different task semantics, page structures, and hosting platforms argues that the backbone-dominance and gain-compounding patterns are properties of

the agents rather than artifacts of a single benchmark, although both environments share the sandboxed, frozen-content design whose external validity Section 5 discusses.

Table 5. WorkArena-L1 Success Rate (%), Mean ±Standard Deviation over 3 Seeds of 165 Instances. The Final Column Shows the Lookahead Gain over Direct Prompting.

Backbone	Direct	Lookahead	Δ
GPT-4	40.0 ±1.8	49.7 ±2.1	+9.7
Llama-3-70B	23.0 ±1.6	29.1 ±1.8	+6.1
GPT-3.5	15.8 ±1.2	20.6 ±1.6	+4.8

5.3. Failure Modes and Cost-Performance Trade-Off

5.3.1. Failure-Mode Distribution

Failed lookahead episodes were classified with the recurring error taxonomy established in multi-environment agent evaluation, distinguishing invalid-format actions, context-limit overruns, and exhausted step budgets, augmented here with an incorrect-early-stop category for episodes that issue a stop action in a wrong state. Table 6 shows clearly stratified profiles. GPT-3.5 spends 31.4% of its failures on actions that do not parse, a mechanical deficiency that scaffolding cannot repair, while GPT-4 rarely emits malformed actions (8.6%) and instead fails by stopping early in an incorrect state (40.1%) or looping until the step budget expires (47.2%). The shift from syntactic to semantic failure with backbone strength explains the strategy-interaction pattern of Section 4.1: reflection and search operate on the semantic failure classes and find little purchase when most errors are mechanical. Manual inspection of a 60-episode sample per backbone supports the automatic classification: looping episodes typically alternate between two page states after a filter or form submission silently fails, and early-stop episodes most often declare success after completing a visually similar yet functionally wrong subtask, such as saving a draft instead of submitting it [45,46].

Table 6. Distribution of Failure Modes (% of Failed WebArena Episodes, Lookahead Configuration, Pooled over Seeds). Each Column Sums to 100.

Failure mode	GPT-4	Llama-3-70B	GPT-3.5
Invalid-format action	8.6	23.5	31.4
Context-limit exceeded	4.1	9.8	12.6
Step-limit / repetitive loop	47.2	41.9	38.7
Incorrect early stop	40.1	24.8	17.3

5.3.2. Cost-Performance Trade-Off

Figure 3 plots success rate against mean inference cost per task for all twelve configurations. Eleven of the twelve points lie on the empirical Pareto frontier; the single dominated configuration is GPT-3.5 with lookahead ($0.078 per task, 9.7%), which Llama-3-70B with ReAct beats on both axes ($0.061, 10.2%). Along the GPT-4 ladder, the marginal cost of one additional success point climbs steeply: upgrading from direct prompting to ReAct buys 2.6 points at roughly $0.04 per point, whereas upgrading from the retry strategy to lookahead buys 2.2 points at roughly $0.40 per point, a near tenfold increase in marginal price. At small budgets the open-weights backbone is attractive, with Llama-3-70B lookahead delivering 12.1% at $0.236 per task, below the cost of even direct GPT-4 prompting ($0.31).

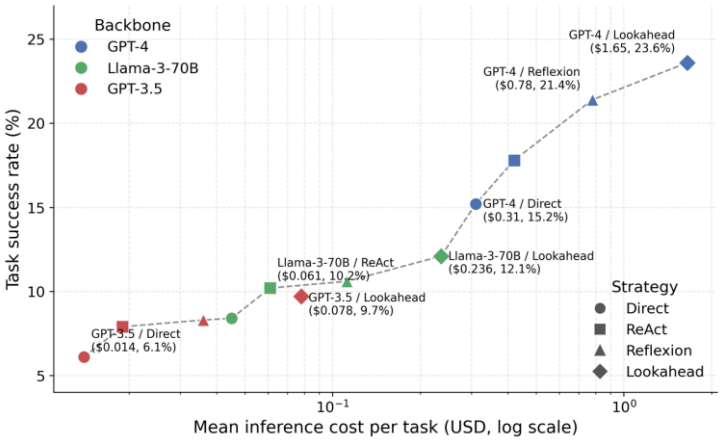

Figure 3. Cost-Performance Frontier of All Twelve Configurations

Success rate versus mean inference cost per task (logarithmic cost axis), with each marker denoting one backbone-strategy configuration and the line tracing the empirical Pareto frontier. GPT-3.5 with lookahead ($0.078, 9.7%) is the only dominated configuration, surpassed on both axes by Llama-3-70B with ReAct ($0.061, 10.2%). Within the GPT-4 family, cost spans $0.31 to $1.65 per task while success rises from 15.2% to 23.6%, and the marginal cost per percentage point increases roughly tenfold between the first and last upgrade step.

6. Discussion

6.1. Findings and Practical Implications

The factorial evidence supports a clear allocation rule: invest in backbone strength before inference-time scaffolding. Backbone choice separates configurations more than any strategy does, strategy gains compound with backbone capability rather than substituting for it, and the failure-mode analysis identifies the mechanism, since scaffolding addresses semantic errors that only stronger backbones make in the first place. The economic analysis sharpens the rule. Once a strong backbone is in place, per-step reasoning is nearly free per point gained, episodic retry is moderately priced, and shallow search is an order of magnitude more expensive per point, so the appropriate stopping rung on the strategy ladder is a budget decision rather than a purely technical one. Cheap backbones invert the calculus: spending their scaffolding budget on a stronger model buys more success than any prompting upgrade examined here, and the dominated GPT-3.5 lookahead configuration illustrates the trap of scaffolding a backbone past its economic niche. The per-site decomposition adds a deployment-facing nuance: teams operating configuration-heavy administrative interfaces can expect the largest returns from scaffolding, while workflows spanning multiple applications gain least and likely require memory or planning mechanisms beyond the per-step scaffolds studied here. The persistent gap to the human reference, more than 54 points even for the best cell, cautions against reading any of these configurations as ready for unsupervised operation; in the near term the realistic deployment pattern is human-supervised execution on the easier site categories, with automatic escalation on the interaction patterns where failure concentrates. The failure profile also suggests an inexpensive engineering lever that is orthogonal to both factors studied: loop detection and forced state summarization target the step-limit failures that account for nearly half of strong-backbone errors, and could plausibly be retrofitted to any of the twelve configurations without altering the backbone or the strategy.

6.2. Limitations

Four limitations bound the conclusions. The study is text-only; visually grounded prompting has shown markedly different behavior under oracle grounding, and extending the factorial design to multimodal observation is a natural next step. The

sandboxed environments freeze content, so robustness to live-web drift and to long-horizon open-web tasks remains untested, and absolute success levels reported here should not be extrapolated to open-web deployment. Programmatic verification occasionally penalizes semantically correct answers phrased unexpectedly, which deflates absolute numbers although it affects all cells equally and leaves relative comparisons intact. Cost figures reflect provider prices and our GPU amortization at the time of experimentation and will shift as pricing evolves, although the ordering of marginal costs across strategies depends on call counts rather than on price levels and should persist under proportional price changes. Future work will extend the grid along three axes: deeper and wider search budgets to map where the cost curve crosses practical limits, trajectory-memory strategies that the present per-step scaffolds exclude, and newer frontier backbones as they stabilize. A further direction is methodological, examining whether the matched-pair statistical machinery used here can be standardized across web-agent leaderboards so that future published numbers become directly comparable.

References

1. S. Zhou, F. F. Xu, H. Zhu, X. Zhou, R. Lo, A. Sridhar, X. Cheng, T. Ou, Y. Bisk, D. Fried, U. Alon, and G. Neubig, "WebArena: A realistic web environment for building autonomous agents," in *Proc. 12th Int. Conf. Learn. Representations (ICLR 2024)*, 2024.
2. X. Deng, Y. Gu, B. Zheng, S. Chen, S. Stevens, B. Wang, H. Sun, and Y. Su, "Mind2Web: Towards a generalist agent for the web," in *Adv. Neural Inf. Process. Syst. 36 (NeurIPS 2023), Datasets and Benchmarks Track*, 2023.
3. H. He, W. Yao, K. Ma, W. Yu, Y. Dai, H. Zhang, Z. Lan, and D. Yu, "WebVoyager: Building an end-to-end web agent with large multimodal models," in *Proc. 62nd Annu. Meeting Assoc. Comput. Linguistics (ACL 2024)*, pp. 6864–6890, 2024.
4. M. Yu and Z. Li, "An Empirical Comparison of Discrete Video Tokenization Schemes for Video Question Answering and Video Captioning," *Artif. Intell. Mach. Learn. Rev.*, vol. 6, no. 2, pp. 27–50, 2025.
5. J. Lai and Z. Li, "A Comparative Empirical Evaluation of Single-Agent and Multi-Agent LLM Prompting Strategies for Automated Formative Feedback in Education," *J. Intell. Eng. Technol.*, vol. 1, no. 2, pp. 29–38, 2026.
6. X. Fu, T. Tang, and C. Luo, "An Empirical Comparison of ReAct, Reflexion, Plan-and-Solve, and Tree-of-Thought Planning Strategies on Financial Question Answering and Numerical Reasoning Tasks," *J. Sci., Innov. Soc. Impact*, vol. 2, no. 3, pp. 23–34, 2026.
7. C. Luo and X. Wang, "Latency-Throughput Tradeoffs of ONNX Runtime, TensorRT-LLM, vLLM, and Triton: An Empirical Comparison on 1B–3B Parameter LLM Inference," *Spectrum Res.*, vol. 6, no. 1, 2026.
8. M. Li and S. Xu, "Trustworthy artificial intelligence in financial decision-making: A systematic review of explainability, fairness, and accountability," *J. Sustain. Policy Pract.*, vol. 2, no. 3, pp. 104–114, 2026.
9. Z. Luo, "Fairness-Aware Credit Evaluation: Bias Detection and Mitigation Techniques for Inclusive Lending Practices," *J. Sustain. Policy Pract.*, vol. 2, no. 2, pp. 78–89, 2026.
10. O. Yoran, S. J. Amouyal, C. Malaviya, B. Bogin, O. Press, and J. Berant, "AssistantBench: Can web agents solve realistic and time-consuming tasks?," in *Proc. 2024 Conf. Empir. Methods Nat. Lang. Process. (EMNLP 2024)*, pp. 8938–8968, 2024.
11. E. Z. Liu, K. Guu, P. Pasupat, T. Shi, and P. Liang, "Reinforcement learning on web interfaces using workflow-guided exploration," in *Proc. 6th Int. Conf. Learn. Representations (ICLR 2018)*, 2018.
12. S. Yao, H. Chen, J. Yang, and K. Narasimhan, "WebShop: Towards scalable real-world web interaction with grounded language agents," in *Adv. Neural Inf. Process. Syst. 35 (NeurIPS 2022)*, pp. 20744–20757, 2022.
13. T. Tang and M. Yu, "A Comparative Empirical Study of Semantic Signal Enhancement Methods for User Interest Features in CTR Prediction: Applicability of TF-IDF Weighting, Sentence-BERT Embeddings, and LDA Topic Fusion," *J. Comput. Innov. Appl.*, vol. 2, no. 1, pp. 165–174, 2024.
14. T. Tang and M. Yu, "A Comparative Evaluation of LLM-Generated Semantic Tags versus Classical Text Features (TF-IDF, LDA, BERT Embeddings) for User-Interest Enrichment in Short-Video Recommendation," *Artif. Intell. Mach. Learn. Rev.*, vol. 5, no. 1, pp. 129–140, 2024.
15. J. Y. Koh, R. Lo, L. Jang, V. Duvvur, M. C. Lim, P.-Y. Huang, G. Neubig, S. Zhou, R. Salakhutdinov, and D. Fried, "VisualWebArena: Evaluating multimodal agents on realistic visual web tasks," in *Proc. 62nd Annu. Meeting Assoc. Comput. Linguistics (ACL 2024)*, pp. 881–905, 2024.
16. Z. Jiang and M. Wang, "Evaluation and Analysis of Chart Reasoning Accuracy in Multimodal Large Language Models: An Empirical Study on Influencing Factors," in *Pinnacle Acad. Press Proc. Ser.*, vol. 3, pp. 43–58, 2025.
17. Y. Tian, "Gradient Boosting-Based Demand Variability Estimation for Improved Safety Stock Calculation in Multi-Echelon Aerospace Spare Parts Inventory," *J. Sci., Innov. Soc. Impact*, vol. 2, no. 2, pp. 175–186, 2026.
18. X. Long, J. Hu, and Z. Ling, "A Comparative Analysis of Telemetry-Driven Anomaly Detection Approaches for Dual-Purpose Operational and Security Optimization in Edge Computing Infrastructure," *J. Comput. Innov. Appl.*, vol. 4, no. 1, pp. 79–88, 2026.

19. A. Drouin, M. Gasse, M. Caccia, I. H. Laradji, M. Del Verme, T. Marty, L. Boisvert, M. Thakkar, Q. Cappart, D. Vazquez, N. Chapados, and A. Lacoste, "WorkArena: How capable are web agents at solving common knowledge work tasks?," in *Proc. 41st Int. Conf. Mach. Learn. (ICML 2024)*, 2024.
20. Y. Chen and J. Hu, "Graph Neural Network-Based Cascading Disruption Path Identification in Multi-Tier Rare Earth Processing Networks," *J. Global Eng. Rev.*, vol. 4, no. 1, pp. 99–112, 2026.
21. L. Boisvert, M. Thakkar, M. Gasse, M. Caccia, T. Le Sellier de Chezelles, Q. Cappart, N. Chapados, A. Lacoste, and A. Drouin, "WorkArena++: Towards compositional planning and reasoning-based common knowledge work tasks," in *Adv. Neural Inf. Process. Syst. 37 (NeurIPS 2024), Datasets and Benchmarks Track*, 2024.
22. L. Zheng, R. Wang, X. Wang, and B. An, "Synapse: Trajectory-as-exemplar prompting with memory for computer control," in *Proc. 12th Int. Conf. Learn. Representations (ICLR 2024)*, 2024.
23. M. Wang, P. Xiao, and M. Yu, "Sparse, Dense, or Hybrid? Comparing Retrieval Strategies for Biomedical Question Answering with Retrieval-Augmented Generation," *J. Sci., Innov. Soc. Impact*, vol. 2, no. 2, pp. 141–152, 2026.
24. B. Zheng, B. Gou, J. Kil, H. Sun, and Y. Su, "GPT-4V(ision) is a generalist web agent, if grounded," in *Proc. 41st Int. Conf. Mach. Learn. (ICML 2024)*, 2024.
25. Y. Chen and T. Tang, "Evaluating Prompt Engineering Strategies for Few-Shot Cyber Threat Intelligence Entity and Relation Extraction from Multi-Source Reports," *J. Sci., Innov. Soc. Impact*, vol. 2, no. 2, pp. 153–164, 2026.
26. J. Lai and Z. Li, "Does an LLM-Based Feedback Agent Move the Needle? An Empirical Study of Student Correctness and Hint Dependency on the ASSISTments 2017 Interaction Logs," *Acad. Nexus J.*, vol. 5, no. 1, 2026.
27. Z. Chen and M. Wang, "Evaluating the Quality of Large Language Model-Generated Explanations in Recommendation Tasks: A Multi-Dimensional Comparative Analysis," *J. Global Eng. Rev.*, vol. 3, no. 1, pp. 54–63, 2025.
28. T. Tang, X. Fu, and C. Luo, "An Empirical Comparison of High-Order Feature Interaction Operators for Conversion Rate Prediction in Sparse, High-Cardinality Message-Ads Traffic: Accuracy, Efficiency, and Offline-Online Consistency," *J. Sci., Innov. Soc. Impact*, vol. 2, no. 3, pp. 12–22, 2026.
29. X. Liu, H. Yu, H. Zhang, Y. Xu, X. Lei, H. Lai, Y. Gu, H. Ding, K. Men, K. Yang, S. Zhang, X. Deng, A. Zeng, Z. Du, C. Zhang, S. Shen, T. Zhang, Y. Su, H. Sun, M. Huang, Y. Dong, and J. Tang, "AgentBench: Evaluating LLMs as agents," in *Proc. 12th Int. Conf. Learn. Representations (ICLR 2024)*, 2024.
30. J. Hu, X. Wang, and J. Lai, "Benchmarking Learned Cardinality Estimation Techniques for Analytical Query Processing in Data Warehouses," *J. Comput. Technol. Appl. Math.*, vol. 3, no. 3, pp. 1–8, 2026.
31. M. Wang and L. Zhu, "Linguistic analysis of verb tense usage patterns in computer science paper abstracts," *Acad. Nexus J.*, vol. 3, no. 3, 2024.
32. M. Wang, Z. Jiang, and S. Zhou, "Cross-cultural semantic differences in emoji usage on social media platforms," *J. Adv. Comput. Syst.*, vol. 4, no. 5, pp. 55–66, 2024.
33. T. Le Sellier de Chezelles, M. Gasse, A. Drouin, M. Caccia, L. Boisvert, M. Thakkar, T. Marty, R. Assouel, S. Omidi Shayegan, L. K. Jang, X. H. Lu, O. Yoran, D. Kong, F. F. Xu, S. Reddy, Q. Cappart, G. Neubig, R. Salakhutdinov, N. Chapados, and A. Lacoste, "The BrowserGym ecosystem for web agent research," in *Proc. Conf. Lang. Model. (COLM 2025)*, 2025.
34. H. Cao, J. Hu, and C. Luo, "Behavioural Feature Analysis for Anomalous Click Detection in Mobile Advertising Environments: Toward In-App Browser-Specific Detection," *J. Comput. Innov. Appl.*, vol. 3, no. 2, pp. 96–105, 2025.
35. Y. Xu, H. Su, C. Xing, B. Mi, Q. Liu, W. Shi, B. Hui, F. Zhou, Y. Liu, T. Xie, Z. Cheng, S. Zhao, L. Kong, B. Wang, C. Xiong, and T. Yu, "Lemur: Harmonizing natural language and code for language agents," in *Proc. 12th Int. Conf. Learn. Representations (ICLR 2024)*, 2024.
36. S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao, "ReAct: Synergizing reasoning and acting in language models," in *Proc. 11th Int. Conf. Learn. Representations (ICLR 2023)*, 2023.
37. N. Shinn, F. Cassano, E. Berman, A. Gopinath, K. Narasimhan, and S. Yao, "Reflexion: Language agents with verbal reinforcement learning," in *Adv. Neural Inf. Process. Syst. 36 (NeurIPS 2023)*, 2023.
38. J. Y. Koh, S. McAleer, D. Fried, and R. Salakhutdinov, "Tree search for language model agents," *arXiv preprint arXiv:2407.01476*, 2024.
39. X. H. Lu, A. Kazemnejad, N. Meade, A. Patel, D. Shin, A. Zambrano, K. Stańczak, P. Shaw, C. J. Pal, and S. Reddy, "AgentRewardBench: Evaluating automatic evaluations of web agent trajectories," *arXiv preprint arXiv:2504.08942*, 2025.
40. H. Lai, X. Liu, I. L. Iong, S. Yao, Y. Chen, P. Shen, H. Yu, H. Zhang, X. Zhang, Y. Dong, and J. Tang, "AutoWebGLM: A large language model-based web navigating agent," in *Proc. 30th ACM SIGKDD Conf. Knowl. Discov. Data Min. (KDD 2024)*, pp. 5295–5306, 2024.
41. P. Sodhi, S. R. K. Branavan, Y. Artzi, and R. McDonald, "SteP: Stacked LLM policies for web actions," in *Proc. Conf. Lang. Model. (COLM 2024)*, 2024.
42. Z. Chen, M. White, R. Mooney, A. Payani, Y. Su, and H. Sun, "When is tree search useful for LLM planning? It depends on the discriminator," in *Proc. 62nd Annu. Meeting Assoc. Comput. Linguistics (ACL 2024)*, pp. 13659–13678, 2024.
43. F. Zhao and T. Tang, "AI-Based Sentiment Analysis for Stock Market Prediction: A Systematic Literature Review," *J. Sustain. Policy Pract.*, vol. 2, no. 3, pp. 115–124, 2026.
44. L. Long and J. Hu, "Multi-Objective Particle Swarm Optimization for Site Selection and Policy Subsidy Maximization of Foreign Renewable Energy Enterprises in the United States," *Artif. Intell. Mach. Learn. Rev.*, vol. 7, no. 2, pp. 54–69, 2026.

45. J. Hu and X. Long, "Graph Learning-Based Behavioral Detection for Software Supply Chain Attacks," *J. Adv. Comput. Syst.*, vol. 4, no. 4, pp. 49–60, 2024.
46. Y. Li, F. Zhao, and J. Hu, "Identifying Cross-Market Risk Contagion Amplifiers via Graph Attention Networks: Empirical Evidence from US Financial Stress Periods," *J. Comput. Innov. Appl.*, vol. 4, no. 1, pp. 164–175, 2026.