OpenSkillEval: Automatically Auditing the Open Skill Ecosystem for LLM Agents

Anonymous ACL submission

Abstract

Skills are increasingly used to improve LLM agents on real-world tasks, yet it remains unclear how agents use skills, how skill quality should be evaluated, and how users should choose skills under cost-performance trade-offs. We present OpenSkillEval, an automatic framework for evaluating both skill-augmented agents and skills. OpenSkillEval dynamically constructs test cases from evolving real-world artifacts across five applications: presentation generation, front-end web design, poster generation, data visualization, and report generation. It also curates community skills for controlled comparison under unified settings. Using over 600 generated tasks and 30 open-source skills, we evaluate state-of-the-art models and agent frameworks. Results show that skill availability does not guarantee effective use, skill benefits depend strongly on the model and framework, and many popular skills fail to consistently outperform skill-free agents. These findings highlight the need for dynamic, task-grounded evaluation and offer practical guidance for skill design, selection, and deployment.

1 Introduction

Recent advances in increasingly capable large language models (LLMs) (OpenAI, 2026b), together with the rapid development of agent client frameworks (Anthropic, 2025a; OpenAI, 2025), have created a promising opportunity to deploy models as autonomous agents for complex downstream tasks. However, because agentic tasks are often open-ended, the overall behavior of an agent can be difficult to predict, and in many challenging settings the agent's intrinsic capability alone may be insufficient for reliable task completion. To better enable models to handle such structured yet complex workflows, developers often formalize personal experience or accumulated best practices into explicit procedures and distill them into structured instructions for agents. These workflow-oriented, formatted instructions are referred to as skills.

Given the promise of skills for augmenting agent capabilities in downstream task completion, strong community participation over the past few months has led to the creation and integration of a large number of skills into the ecosystem for LLM agents. However, this rapid growth has also introduced several important challenges. First, it remains unclear how different agent frameworks perform on downstream tasks in general, and how they interact with the added skill during execution. Lack of systematic evaluation makes it difficult to assess their actual effectiveness. As a result, users may struggle to choose suitable agents for downstream tasks. If an agent lacks the capability to properly execute a provided skill, the practical value of skill augmentation can be greatly reduced. On the other hand, if an agent is already sufficiently strong to solve the task on its own, introducing skills may bring only limited benefit while still increasing execution cost. Second, individually distilled skills may reflect only partial or subjective experience, which can limit their generalizability. As the number of skills grows, it also becomes increasingly unclear how to select the appropriate skills for prompting, especially when users must balance performance and cost. In addition, the repeated submission of redundant or low-quality skills imposes substantial maintenance overhead on the community and contributes to the bloating of the ecosystem.

To address these challenges, we propose **OpenSkillEval**, an automatic evaluation framework for both skill-augmented agent systems and the skills themselves in downstream applications. Instead of relying on static benchmarks, OpenSkillEval dynamically generates test cases that continuously reflect evolving user needs, enabling a more realistic, timely, and comprehensive evaluation setting. Based on these dynamically constructed task instances, we evaluate the effectiveness and effi-

ciency of different models and agent frameworks, both with and without skill augmentation, across diverse downstream tasks. Moreover, by collecting skill sets from the open-source community for each target application, OpenSkillEval enables controlled comparisons of different skills under the same task setting, making it possible to analyze their relative quality, robustness, and transferability. Across five major real-world application categories, presentation generation, front-end web design, poster generation, data visualization, and report generation, and more than 600 tasks, we evaluate a range of state-of-the-art agent architectures and derive several key findings: **1)** Skill availability does not guarantee effective usage: across different client frameworks, agents explicitly read the provided skill in only about 48% of cases on average under a realistic skill-access setting. This suggests that in complex and noisy online environments, even carefully designed skills may be underutilized or ignored by agents (Section 3.1). **2)** Skill benefits depend strongly on the agent framework and underlying model: a weaker model can sometimes match a stronger one when paired with suitable skills. However, when the underlying model is intrinsically weak at solving a task, simply adding skills does not reliably produce gains (Section 3.2). **3)** Skill quality varies substantially across open-source skills: richer and well-designed priors can help agents trade increased input scaling for improved performance, but many popular skills still fail to outperform base agents while introducing additional cost. Our analysis further provides practical takeaways for skill format (Section 3.3).

2 OpenSkillEval Framework

To evaluate the rapidly expanding open-skill ecosystem, we design OpenSkillEval as a sustainable framework for real-world downstream tasks. It consists of three pipelines: automatic test case generation, which builds realistic task instances from user needs (Section 2.1); skill collection and organization, which curates community skills by task category (Section 2.2); and evaluation, which assesses both final artifact quality and agent trajectories (Section 2.3). Together, these components enable dynamic auditing at both agent-level and skill-level. Figure 1 provides an overview.

2.1 Automatic Case Generation

To ensure practical relevance, we evaluate five common downstream tasks: presentation generation, front-end web design, poster generation, data visualization, and report generation. These tasks are widely used, require multi-step reasoning and tool use, and produce concrete artifacts suitable for direct evaluation. We construct test cases with an **artifact-driven case generation strategy**. Instead of manually writing instructions, we start from high-quality artifacts and infer the user requests that could have produced them, yielding instances that better reflect real user needs and expected outputs. For each task category, the pipeline has three stages. First, we collect artifacts or source materials (S) from diverse repositories. Second, LLMs extract structured task specifications (T) and natural-language instructions (I) under predefined schemas, capturing the inferred user intent, constraints, and expected outputs. Third, we validate each instance for consistency and compatibility with the source content. We next detail the source selection and extraction process for each task.

Presentation Generation. For PPT generation, we collect up-to-date, information-dense webpages and documents, such as benchmark leaderboards, industry reports, and academic papers, that naturally support slide-deck summarization. Each source is crawled, snapshot-preserved, and processed by an LLM-based pipeline to produce slide-level task specifications with explicit per-slide content goals, forming realistic presentation requests grounded in real-world materials.

Front-end Web Design. For web design, we use existing websites as target artifacts. We automatically curate high-quality public websites from design award platforms and product discovery sites, organizing them by site type and domain. For each site, we capture its rendered layout, navigation structure, and interactive components, then use LLM-based reverse engineering to convert these observations into structured design specifications.

Poster Generation. For poster generation, we collect source content reflecting common poster needs, including data-report visualization, product promotion, event advertising, and social advocacy. These materials span domains such as technology, health, environment, and business, and are organized by use scenario. LLMs then transform each source into a poster task specification defining the target audience, core message, and key content blocks.

Data Visualization. For data visualization, we first survey high-quality examples from open data portals and scientific publications to derive a taxonomy of visualization types, domains, and analytical

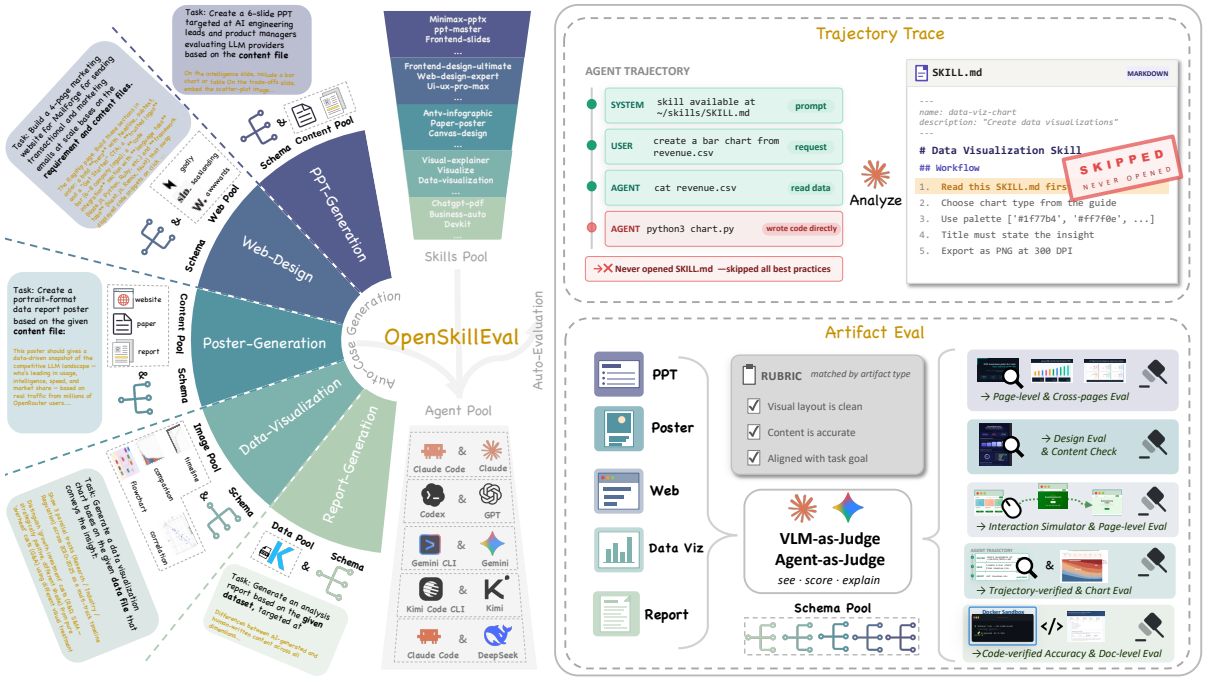


Figure 1: Overview of the OpenSkillEval framework.

goals. We then sample task configurations from this taxonomy and use LLMs to instantiate concrete task specifications. For each specification, we generate a matching data table consistent with the intended visualization and analysis objective, enabling end-to-end evaluation.

Report Generation. For report generation, we build from public real-world tabular datasets covering domains such as e-commerce, finance, healthcare, education, and technology. Based on these datasets, LLMs construct task specifications along dimensions such as report type, required sections, and analysis goals. Each specification is grounded in the dataset by linking requested analyses and report components to concrete data columns.

For validation, a verifier LLM checks whether T and I are information-complete and well-grounded in S, filtering out instances that are inconsistent with the source content. Because the pipeline is automated, the benchmark can be continuously refreshed as source materials evolve, keeping pace with changing real-world needs. In the current version, we use Claude-Opus and GPT-5.2 as primary generators, producing 677 task instances. Their distribution is shown in Figure 8, with task schemas and case studies provided in Appendix A.4.

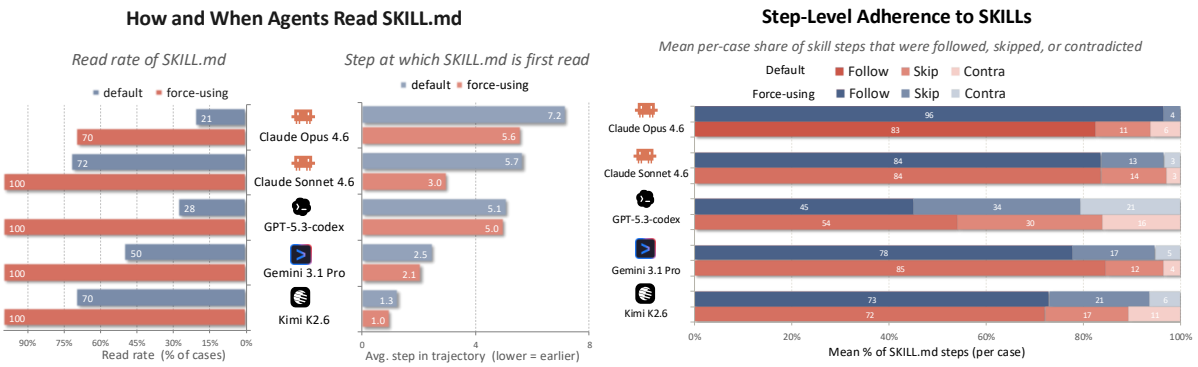
2.2 Skills Collection

Since the open-source skill ecosystem is actively evolving, we collect task-relevant skills from multiple public repositories, including clawhub.ai, skills.sh, openskills.space, and skillsmp.com. Because community skills vary widely in quality, we do not include all retrieved skills. Instead, under a cost-conscious setting, we filter for skills with relatively high community adoption using download counts as a proxy. This yields 30 skills for evaluation, with detailed statistics reported in Table 4.

2.3 Automatic Evaluation Pipeline

To evaluate skill-augmented agents, we design an automatic pipeline from two complementary perspectives: **trajectory trace analysis**, which examines the execution process, and **artifact analysis**, which assesses final output quality. For trajectory analysis, we introduce an agent-as-judge procedure that decomposes each skill into intended workflow steps and compares them with the actual trajectory at a fine-grained level. This allows us to assess whether the agent invokes the skill at the right stage and follows the prescribed workflow.

Artifact analysis evaluates whether final outputs meet task requirements in realistic settings. For each task category, we design task-specific criteria inspired by PPTEval (Zheng et al., 2025b), GenEval (Hong et al., 2025), and WebArena (Zhou et al., 2023), while adapting them to each downstream application. Across tasks, we assess shared dimensions such as *completeness*, *content quality*, and *visual design*. We further introduce spe-



(a) (b)

Figure 2: Trajectory analysis of how agents access and follow provided skills. **(a)** Statistics of `SKILL.md` access under the default and *force-using* settings, including the proportion of cases in which the skill is explicitly read and the average trajectory step at which it is first accessed. **(b)** Step-level adherence to skill workflows after skill access, showing the mean proportion of prescribed steps that are followed, skipped, or contradicted across agents.

cialized procedures for targeted assessment. For web design, we combine screenshot-based visual evaluation with agent-based interaction to assess functional completeness. For report generation and data visualization, we explicitly evaluate *data accuracy*: reports are checked through code-based consistency tests between reported values, conclusions, and underlying data, while visualizations are assessed through artifact inspection and trajectory analysis to verify correct data usage and representation. Detailed evaluation process and matrices are shown in Appendix A.3. We additionally conduct human evaluation to validate both generated tasks and automatic evaluation results (Appendix A.5).

3 Experimental Results

Using OpenSkillEval, we evaluate state-of-the-art agent systems. Detailed settings are shown in Appendix A.2. We analyze skill-augmented agents from three perspectives: trajectory behavior (Section 3.1), overall agent performance (Section 3.2), and skill quality differences (Section 3.3).

3.1 Trajectory Trace Analysis

Before large-scale evaluation, we conduct trajectory analysis to examine how agents use skills in practice. We place each target skill in the initialization path of each CLI-based framework, mirroring the common setting where users make skills accessible to agents. Agents then complete the generated tasks, and we analyze their execution traces to determine whether and how the injected skill is used.

We find that skills are often under-used by default. As shown in Figure 2a, when sampling 100 task instances per agent across task categories, agents explicitly read the corresponding `skill.md` file in only about 48% of cases on average. Even strong models such as Claude Opus 4.6 read the skill in only around 20% of cases. **This indicates that simply placing a skill in an accessible environment does not ensure that agents actively use it.** To enable controlled evaluation of the potential benefits of skill augmentation, we further consider a *force-using* setting that follows the intended skill-usage strategy recommended by agent frameworks, where the task instruction explicitly directs the agent to invoke the target skill. This substantially increases the average skill read rate to 94% and encourages earlier access, reducing the average number of pre-read steps from 4.4 to 3.3.

However, explicit invocation does not eliminate agent autonomy. Fine-grained analysis shows that, even after reading a skill, agents may skip prescribed steps (*Skip*) or deviate from the intended procedure (*Contra*), as shown in Figure 2b. This autonomy is especially notable for Claude Opus 4.6. Although force-using improves its skill access, its overall usage rate remains lower than several other agents (Figure 2a). Further analysis reveals strong task-level variation: its skill read rate remains below 50% for data visualization and report generation, but exceeds 95% for tasks such as presentation generation. This pattern suggests a more selective style of skill usage, in which the agent appears to consult skills only when it deems them especially helpful for the task. Supporting this interpretation, Claude Opus spends more steps analyzing the task before first accessing `skill.md`, as shown in Figure 2a. This resembles deliberate decision-making patterns observed in retrieval-augmented generation (Ying et al., 2024b) and real-world Claude skill usage (Seleznov, 2026). **Together, these findings**

Metric	Claude Opus 4.6	Claude Sonnet 4.6	GPT-5.5	GPT-5.3-codex	GPT-5.2	Gemini 3.1 Pro	DeepSeek V4	Kimi K2.6	MiniMax M2.7
Presentation Generation *ppt-generation*									
Content quality	$\mathbf{4.48 \pm 0.35}$	4.44 ± 0.41	4.41 ± 0.39	3.55 ± 0.52	3.74 ± 0.49	3.78 ± 0.43	4.33 ± 0.33	4.14 ± 0.53	4.18 ± 0.42
Visual design	3.98 ± 0.30	3.92 ± 0.27	$\mathbf{4.01 \pm 0.28}$	3.04 ± 0.46	3.23 ± 0.50	2.99 ± 0.64	3.79 ± 0.29	3.57 ± 0.52	3.73 ± 0.36
Completeness	$\mathbf{4.68 \pm 0.54}$	4.63 ± 0.67	4.66 ± 0.70	4.07 ± 1.08	4.46 ± 0.74	4.24 ± 0.67	4.67 ± 0.58	4.53 ± 0.75	4.44 ± 0.77
Fidelity	4.51 ± 0.53	4.31 ± 0.52	$\mathbf{4.88 \pm 0.35}$	4.02 ± 1.13	4.84 ± 0.38	4.61 ± 0.51	4.20 ± 0.49	4.46 ± 0.72	4.18 ± 0.61
Task avg.	*4.41*	*4.33*	***4.49***	*3.67*	*4.07*	*3.90*	*4.25*	*4.17*	*4.13*
Poster Generation *poster-generation*									
Content quality	4.56 ± 0.71	4.37 ± 0.75	$\mathbf{4.60 \pm 0.71}$	4.42 ± 0.91	4.49 ± 0.76	4.43 ± 0.70	4.39 ± 0.74	4.49 ± 0.73	4.20 ± 0.81
Visual design	$\mathbf{4.00 \pm 0.60}$	3.72 ± 0.77	3.82 ± 0.58	3.19 ± 0.76	3.11 ± 0.83	3.39 ± 0.84	3.55 ± 0.75	3.48 ± 0.84	3.10 ± 0.88
Completeness	$\mathbf{4.12 \pm 0.53}$	3.98 ± 0.66	3.97 ± 0.63	3.44 ± 0.79	3.43 ± 0.84	3.41 ± 0.80	3.90 ± 0.65	3.68 ± 0.74	3.35 ± 0.85
Task avg.	***4.23***	*4.02*	*4.13*	*3.68*	*3.67*	*3.74*	*3.94*	*3.88*	*3.55*
Data Visualization *data-visualization*									
Insight expression	$\mathbf{4.40 \pm 0.62}$	4.22 ± 0.70	4.04 ± 0.77	2.79 ± 1.12	3.15 ± 1.10	3.66 ± 1.00	3.96 ± 0.73	3.82 ± 0.85	3.31 ± 1.03
Visual quality	$\mathbf{4.06 \pm 0.62}$	3.99 ± 0.71	3.54 ± 0.79	3.13 ± 0.86	2.92 ± 0.93	3.39 ± 0.92	3.59 ± 0.72	3.52 ± 0.82	3.30 ± 0.91
Completeness	$\mathbf{4.80 \pm 0.42}$	4.67 ± 0.57	4.58 ± 0.67	2.87 ± 1.35	3.46 ± 1.25	4.03 ± 1.03	4.43 ± 0.71	4.27 ± 0.88	3.62 ± 1.13
Data accuracy	$\mathbf{4.98 \pm 0.15}$	4.93 ± 0.25	4.94 ± 0.26	4.25 ± 1.00	4.78 ± 0.59	4.93 ± 0.27	4.95 ± 0.21	4.92 ± 0.30	4.82 ± 0.41
Task avg.	***4.56***	*4.45*	*4.28*	*3.26*	*3.58*	*4.00*	*4.23*	*4.13*	*3.76*
Report Generation *report-generation*									
Content quality	4.20 ± 0.77	$\mathbf{4.43 \pm 0.71}$	4.03 ± 0.72	2.53 ± 0.85	3.25 ± 0.97	2.82 ± 0.71	3.90 ± 0.86	4.29 ± 1.00	3.35 ± 0.81
Visual quality	4.60 ± 0.66	$\mathbf{4.68 \pm 0.67}$	4.52 ± 0.62	3.81 ± 0.90	3.89 ± 0.98	3.99 ± 0.71	4.38 ± 0.97	4.35 ± 1.09	4.11 ± 0.92
Completeness	4.77 ± 0.52	$\mathbf{4.81 \pm 0.50}$	4.79 ± 0.53	3.21 ± 1.19	4.01 ± 1.14	3.14 ± 0.87	4.54 ± 0.72	4.62 ± 1.00	4.29 ± 0.78
Data accuracy	4.87 ± 0.37	4.74 ± 0.54	$\mathbf{4.95 \pm 0.24}$	4.66 ± 0.88	4.87 ± 0.47	4.74 ± 0.60	4.76 ± 0.51	4.60 ± 0.98	4.50 ± 0.81
Fidelity	4.58 ± 0.58	4.44 ± 0.65	$\mathbf{4.87 \pm 0.36}$	4.46 ± 0.92	4.82 ± 0.49	4.28 ± 0.89	4.21 ± 0.74	4.31 ± 1.00	3.89 ± 0.93
Task avg.	*4.60*	*4.62*	***4.63***	*3.73*	*4.17*	*3.79*	*4.36*	*4.43*	*4.03*
Front-end Web Design *web-design*									
Visual design	4.62 ± 0.48	$\mathbf{4.75 \pm 0.42}$	4.71 ± 0.53	4.12 ± 0.56	4.44 ± 0.58	4.20 ± 0.84	4.56 ± 0.48	4.10 ± 1.30	4.43 ± 0.63
Responsive	4.34 ± 0.47	4.21 ± 0.51	$\mathbf{4.56 \pm 0.55}$	4.16 ± 0.59	4.18 ± 0.60	4.06 ± 0.61	4.38 ± 0.48	4.02 ± 1.08	4.19 ± 0.60
Navigation pass rate	4.96 ± 0.30	4.97 ± 0.23	4.96 ± 0.33	4.82 ± 0.59	$\mathbf{4.98 \pm 0.24}$	4.96 ± 0.26	4.96 ± 0.27	4.70 ± 1.13	4.93 ± 0.44
Interaction pass rate	4.92 ± 0.33	$\mathbf{4.93 \pm 0.33}$	4.91 ± 0.46	4.73 ± 0.66	4.82 ± 0.51	4.74 ± 0.68	4.91 ± 0.38	4.59 ± 1.15	4.84 ± 0.51
Data display pass rate	4.95 ± 0.22	4.94 ± 0.21	4.95 ± 0.32	4.72 ± 0.71	$\mathbf{4.96 \pm 0.14}$	4.93 ± 0.23	4.94 ± 0.21	4.70 ± 1.05	4.90 ± 0.27
Task avg.	*4.74*	*4.75*	***4.80***	*4.47*	*4.66*	*4.55*	*4.73*	*4.40*	*4.63*
Overall avg.	**4.51**	4.43	4.47	3.76	4.03	4.00	4.30	4.20	4.02

Table 1: Per-task evaluation scores by agent (1–5 scale). Each cell shows mean±std across cases.

show that agents exhibit nontrivial autonomous decision-making in how they use skills.

Since such agent autonomy is itself an important part of how skill augmentation functions in practice, we preserve this behavior in our evaluation. Accordingly, in the subsequent experiments, we evaluate skill augmentation under both *no-skills* and *force-using skills* settings, and assess performance primarily based on the final output artifacts.

3.2 How Different Agents Perform

We compare agent systems across downstream tasks under the settings in Section 2.3. Overall, Claude 4.6 with Claude Code and GPT-5.5 with Codex achieve the strongest and most stable performance across the five task categories (Table 1), indicating better adaptability across skill-use settings. By contrast, GPT-Codex performs below its coding-oriented reputation, possibly due to weaker agentic capability in open-ended generation tasks. Our skill-level analysis further shows that when the base agent is limited, even high-quality skills cannot fully realize their benefits (Section 3.3).

Across tasks, several patterns emerge. **1)** Presentation and poster generation are the most challenging categories, especially in visual design and layout quality, where average scores are generally below 4. Common failures include unbalanced composition, over-compressed content, and overlapping elements. As discussed later, detailed task-specific skills such as `ppt-master` can partially mitigate these issues (Section 3.3). **2)** Front-end web design shows relatively strong performance, particularly in interactive functionality, where many agents can already produce usable interfaces. However, the results remain far from deployment quality: responsive design is consistently weaker, and visual issues such as improper element sizing and unbalanced spacing remain common. **3)** For data visualization, most models can reliably use structured data files and produce charts with high data accuracy. Their main weakness lies in analytical framing: generated visualizations often fail to highlight key variable relationships or communicate clear insights. **4)** For report generation, HTML rendering often yields visually polished reports, and agents score highly on visual quality. However, content quality and fidelity remain weaker. Reports are often descriptive rather than analytically rigorous, frequently missing significance testing, robustness checks, or explicit comparative analysis. **These suggest that even the best agents still remain noticeably below the ideal setting in which generated artifacts can be directly used**

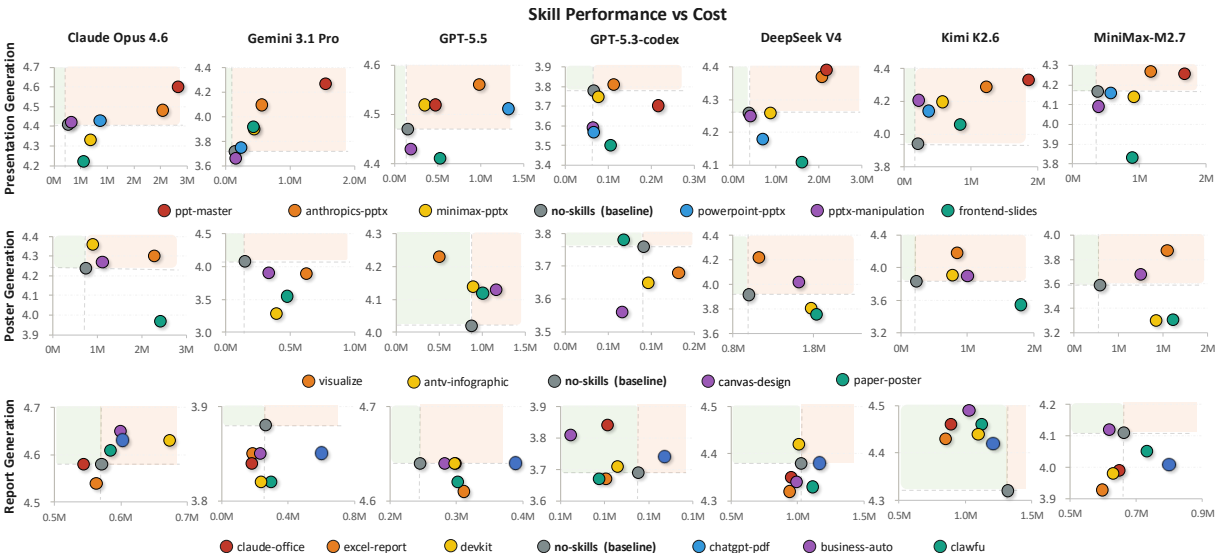


Figure 3: Skill performance versus cost across tasks and agent systems. The full result is shown in Table 9.

without intervention, indicating that effective human oversight is still necessary in practice.

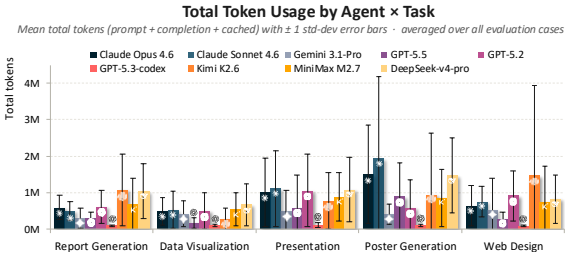


Figure 4: Token usage across agents and tasks.

Beyond performance, we further analyze token usage across agents (Figure 4). Codex framework consistently shows the lowest token consumption, consistent with prior findings (Du et al., 2025), suggesting a design emphasis on efficiency. Notably, GPT-5.5 maintains this efficiency while achieving strong performance, offering a favorable efficiency-performance trade-off compared with other model families and even using fewer tokens than GPT-5.2 in the same series. Claude 4.6 shows moderate and stable consumption overall, except in poster generation, where token usage increases substantially, partly due to poster-specific skills (Section 3.3). In contrast, the Kimi series exhibits weaker execution stability, with large variance and frequent abnormal token consumption; in some cases, Kimi K2.6 enters loops and runs until timeout.

3.3 How Different Skills Perform

We compare skill effects across tasks and agents. As shown in Figure 3 (full results in Figure 9), skill augmentation varies substantially by model, task, and skill. GPT-5.3-codex, for example, often uses skills less effectively and may perform worse with skills, indicating that benefits depend on the agent's ability to execute them. Still, some skills yield consistent gains: `ppt-master` and `anthropics-pptx` improve presentation generation, while `visualize` is effective for poster generation. These gains come at substantially higher inference cost: skill-augmented runs typically consume 3-5× more tokens than no-skill runs. We organize the following analysis into two parts: visually dominated artifact-generation tasks, including poster generation, presentation generation, and front-end web design, where skills mainly affect visual language, layout, and style control; and reasoning or analysis-intensive tasks, including data visualization and report generation, where skills primarily guide problem decomposition and analytical reasoning.

For Visually Oriented Artifact Generation such as presentations and posters, design language is central to output quality. Beyond rubric-based evaluation, we therefore analyze stylistic diversity using CSD-ViT-L style embeddings (Somepalli et al., 2024) and the Vendi Score, computed within each skill group. We control sample size across settings, compare each skill against a matched *no-skills* baseline, and report a pooled multi-skill condition (*pool*) to capture cross-skill variation. Results are shown in Figures 10a and 10b (for front-end web design result in Figure 11, we do not conduct further diversity analysis, for the overall diversity effect of skills is small (less than 0.02)). We find that **skill usage does not necessarily increase stylistic diversity, even across skills**. In presentation generation, some skills such

Skill	Key Assets	Key Constraints	A	C	Obs. O	Obs. D	Obs. E	Obs. Div
Presentation Generation *ppt-generation*								
ppt-master	101 layouts + 33 charts + 640 icons + 39 references	25 MUST + 10 NEVER	5	5	4.27±	3.91±	4.42±	-0.46
anthropics-pptx	2 references (pptxgenjs, editing) + 78 inline design lines	1 MUST + 3 NEVER	3	2	4.20±	3.65±	4.46±	+1.12
minimax-pptx	5 references (incl. design system) + 8 inline design lines	9 MUST + 1 NEVER	2	2	4.09±	3.47±	4.40±	+2.49
powerpoint-pptx	3 references (slides, charts, design) + 21 inline design lines	2 MUST	1	1	4.03±	3.25±	4.47±	+0.27
frontend-slides	Viewport-base CSS + 12-style preset library	13 MUST + 7 NEVER	2	5	3.96±	3.52±	4.24±	-1.65
pptx-manipulation	Unpack/repack tutorial + 16 inline design lines	—	1	0	4.02±	3.29±	4.43±	+0.08
no-skills	—	—	0	0	4.04±	3.20±	4.54±	—
Poster Generation *poster-generation*								
visualize	Mandatory skeleton + 9 references + 98 inline design lines	117 MUST + 28 NEVER	4	5	4.05±	3.93±	4.11±	-2.75
canvas-design	27 inline design lines	15 MUST + 5 NEVER	1	5	3.91±	3.56±	4.09±	-2.08
antv-infographic	External AntV template library	DSL grammar	1	2	3.77±	3.47±	3.92±	-0.83
paper-poster	External LaTeX tcbposter class + 256 inline design lines	21 MUST + 15 NEVER	2	5	3.67±	2.88±	4.07±	-3.07
no-skills	—	—	0	0	3.95±	3.56±	4.14±	—

Table 2: Quantifying skills along two dimensions and relating them to quality and diversity. **Asset score** (A, 0–5) measures design priors packaged with a skill, based on templates, reference documents, and design-related lines in `SKILL.md`. **Constraint score** (C, 0–5) measures procedural constraints, based on `MUST`/`NEVER` directives and structural workflow locks. **Obs.** O is the overall score; **Obs.** D is the design score taken from the judge; **Obs.** E is content-oriented effectiveness, computed as the mean of Completeness and Fidelity for presentations and the mean of Content and Completeness for posters; **Obs. Div** is the visual-diversity score. More results are shown in Table 5

as `anthropics-pptx` and `minimax-pptx` increase diversity, while others such as `frontend-slides` reduce it. In poster generation, skill augmentation generally decreases diversity, suggesting that poster skills impose stronger stylistic constraints. The diversity effect also varies across models, with Pearson correlations of 0.52 for presentations and 0.28 for posters, indicating that stylistic outcomes depend on the interaction between skills and agents.

To understand how skill design affects output quality and stylistic diversity, we analyze the visual-generation skills in Table 2 along two dimensions. **Design priors** capture reusable visual experience encoded in the skill, including templates, reference documents, design assets, and design-related content in `SKILL.md`. **Procedural constraints** capture how strongly the skill prescribes the generation process through explicit directives, such as `MUST`/`NEVER` instructions and other workflow locks. We operationalize both dimensions by extracting countable signals with rule-based heuristics and manually mapping them to coarse 0-5 scores for trend-level comparison across skills.

In terms of output quality, skill design shows a clear relationship with observed performance. Skills almost always improve visual-design scores over the *no-skills* setting, especially in presentation generation, and the gains generally increase with richer *design priors*. This suggests that well-designed visual priors can raise the lower bound of artifact quality by providing concrete design guidance. However, content-oriented metrics such as *Completeness* and *Fidelity* may slightly decline, since skills introduce additional instructions and assets that increase the burden on layout planning and content allocation. More importantly, strong visual design alone is insufficient: useful skills must also remain adaptable across usage scenarios. Some skills impose visually beneficial but content-restrictive constraints. For example, `frontend-slides` enforces hard density caps and strong layout constraints, producing disciplined layouts but often omitting content when source materials are dense, thereby reducing completeness and overall scores. Together, these results suggest an **asset scaling effect: richer and better-designed visual priors tend to improve overall task performance.** Ideally, if a skill provides broad, high-quality design coverage for target use cases, agents can achieve strong performance by operating within that prior space. For example, with `ppt-master`, which covers 22 presentation genres, Kimi K2.6 and Gemini 3.1 improve from around 3.9 to 4.3 and from 3.7 to 4.3, respectively, approaching raw Claude-Opus performance.

In terms of stylistic diversity, although different forms of constraints are not perfectly comparable, we observe a broad negative relationship between constraint strength and output diversity. Skills with stronger procedural constraints tend to produce

more visually consistent but less diverse outputs. This pattern is particularly clear in presentation generation: `frontend-slides` and `ppt-master`, which have the highest constraint scores, also exhibit some of the lowest observed diversity scores.

For Analytical and Reasoning-Intensive Tasks such as report generation and data visualization, require structured reasoning, careful data interpretation, and sound analysis procedures. We therefore emphasize reasoning-related indicators beyond overall scores. For report generation, we focus on *data accuracy* and *fidelity*, assessing whether claims are grounded in the data. For data visualization, we emphasize whether the chart clearly communicates the intended insight.

As shown in Figure 5, we find two main patterns. **1)** Existing skills provide limited benefits for tasks centered on reasoning and analytical decision-making. In report generation, most skills perform only on par with or slightly above the *no-skills* baseline, while `business-auto` is nearly equivalent to *no-skills* because it adds little analytical guidance. Qualitatively, generated reports often remain superficial, lacking significance testing, deeper validation, or rigorous support. A similar pattern appears in data visualization: agents can often produce charts, but fail to surface key variable relationships or communicate the intended insight. This reflects the current skill ecosystem, where many skills provide tool-level assistance rather than analytical priors, structured workflows, or reasoning norms. As a result, success still depends mainly on the model's own analytical ability. **2)** Overall, skill augmentation yields little improvement on these tasks, with an average score gain below 0.04. The main exception is Kimi, where several skill-augmented settings outperform *no-skills*; trace inspection suggests this is partly because Kimi is more prone to looping without skills, rather than because the skills provide large intrinsic gains.

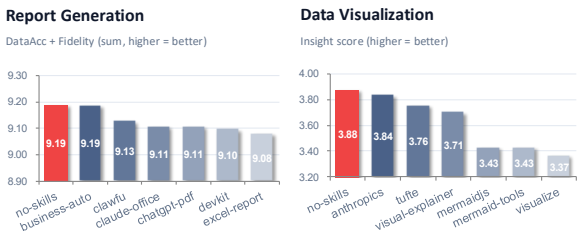


Figure 5: Comparison of skill-augmented and *no-skills* settings on reasoning intensive tasks.

Cost Analysis. Skill augmentation usually increases generation cost (Figure 3), partly because skills add extra input content. This is especially evident for asset-rich skills such as `ppt-master` and `visualize`, whose reference materials increase prompt length and token usage, indicating that asset scaling is accompanied by cost scaling. Some skills further increase cost by enforcing iterative procedures. For example, `paper-poster` requires LaTeX compilation and rule-based checking, causing some models, especially Claude 4.6, to enter longer execution loops. This contributes to the larger variance in Figure 4. In extreme cases, output-token cost exceeds the *no-skills* setting by more than 10×.

4 Related Work

Agent skills and procedural augmentation. Agent Skills (Anthropic, 2025b) package procedural expertise as portable `SKILL.md`-based artifacts. Recent work studies automatic skill construction, discovery, and refinement (Xu and Yan, 2026; Alzubi et al., 2026; Yang et al., 2026; Zheng et al., 2025a), while benchmarks such as SkillsBench (Li et al., 2026), PinchBench (PinchBench Contributors, 2026), and WildClawBench (Ding et al., 2026) evaluate agent–skill combinations. In contrast, we focus on downstream application tasks and compare community skills under unified settings.

Benchmarks for LLM agents. Agent benchmarks cover software engineering (Jimenez et al., 2023), web navigation (Zhou et al., 2023), and reasoning (Liu et al., 2023), but static task can saturate and risk contamination (Cao et al., 2025; Ying et al., 2024a). Dynamic benchmarks release fresh tasks over time (Tang et al., 2025; White et al., 2025; Jain et al., 2024). We extend this to skill-augmented agents, enabling controlled skill comparison while measuring both effectiveness and efficiency.

5 Conclusion

We present OpenSkillEval, an automatic framework for evaluating skill-augmented LLM agents and open-source skills on real-world downstream tasks. By dynamically constructing task instances from evolving artifacts, collecting community skills, and evaluating both trajectories and final outputs, OpenSkillEval enables scalable analysis of skill effectiveness. Our experiments show that skill availability does not guarantee effective use, skill benefits depend strongly on model and agent framework, and many popular skills yield limited or inconsistent gains. We hope OpenSkillEval provides practical guidance for selecting agents and skills, while informing the design and maintenance.

6 Limitations

The open skill ecosystem for LLM agents is still evolving rapidly. In this work, we build an automatic evaluation framework to make large-scale and continuously updated assessment feasible. However, this design also comes with several limitations. First, due to practical cost constraints, the current version of OpenSkillEval does not cover more exhaustive set of available skills. Second, while automation enables scalability and reproducibility, it inevitably abstracts away part of the human-agent interaction process that may matter in real deployment settings. More broadly, our current evaluation primarily analyzes skill effectiveness from two perspectives: the agent's execution process and the quality of the final output artifact. Although we provide taskaways for skills formalization, we do not directly evaluate skills in isolation. This is mainly because the utility of a skill depends strongly on how different models and agent frameworks interpret and use it.

References

Salaheddin Alzubi, Noah Provenzano, Jaydon Bingham, Weiyuan Chen, and Tu Vu. 2026. Evoskill: Automated skill discovery for multi-agent systems.

Anthropic. 2025a. Claude code by anthropic | ai coding agent, terminal, ide. https://www.anthropic.com/claude-code.

Anthropic. 2025b. Equipping agents for the real world with agent skills. https://www.anthropic.com/engineering/equipping-agents-for-the-real-world-with-agent-skills.

Anthropic. 2026. System card: Claude Opus 4.6. Technical report, Anthropic.

Yixin Cao, Shibo Hong, Xinze Li, Jiahao Ying, Yubo Ma, Haiyuan Liang, Yantao Liu, Zijun Yao, Xiaozhi Wang, Dan Huang, Wenxuan Zhang, Lifu Huang, Muhao Chen, Lei Hou, Qianru Sun, Xingjun Ma, Zuxuan Wu, Min-Yen Kan, David Lo, Qi Zhang, Heng Ji, Jing Jiang, Juanzi Li, Aixin Sun, Xuanjing Huang, Tat-Seng Chua, and Yu-Gang Jiang. 2025. Toward generalizable evaluation in the llm era: A survey beyond benchmarks.

DeepSeek-AI. 2026. Deepseek-v4: Towards highly efficient million-token context intelligence.

Shuangrui Ding, Xuanlang Dai, Long Xing, Shengyuan Ding, Ziyu Liu, Jingyi Yang, Penghui Yang, Zhixiong Zhang, Xilin Wei, Yubo Ma, Haodong Duan, Jing Shao, Jiaqi Wang, Dahua Lin, Kai Chen, and Yuhang Zang. 2026. Wildclawbench.

Zheng Du, Hao Kang, Song Han, Tushar Krishna, and Ligeng Zhu. 2025. Ockbench: Measuring the efficiency of llm reasoning. *arXiv preprint arXiv:2511.05722*.

Google. 2025. Gemini CLI. https://github.com/google-gemini/gemini-cli. Accessed: 2026-05-02.

Google DeepMind. 2026. Gemini 3.1 pro model card. https://deepmind.google/models/model-cards/gemini-3-1-pro/.

Harbor Framework Team. 2026. Harbor: A framework for evaluating and optimizing agents and models in container environments.

Shibo Hong, Jiahao Ying, Haiyuan Liang, Mengdi Zhang, Jun Kuang, Jiazheng Zhang, and Yixin Cao. 2025. Frabench and ufeval: Unified fine-grained evaluation with task and aspect generalization.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. 2024. Livecodebench: Holistic and contamination free evaluation of large language models for code.

Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. 2023. Swe-bench: Can language models resolve real-world github issues?

Xiangyi Li, Wenbo Chen, Yimin Liu, Shenghan Zheng, Xiaokun Chen, Yifeng He, Yubo Li, Bingran You, Haotian Shen, Jiankai Sun, Shuyi Wang, Binxu Li, Qunhong Zeng, Di Wang, Xuandong Zhao, Yuanli Wang, Roey Ben Chaim, Zonglin Di, Yipeng Gao, Junwei He, Yizhuo He, Liqiang Jing, Luyang Kong, Xin Lan, Jiachen Li, Songlin Li, Yijiang Li, Yueqian Lin, Xinyi Liu, Xuanqing Liu, Haoran Lyu, Ze Ma, Bowei Wang, Runhui Wang, Tianyu Wang, Wengao Ye, Yue Zhang, Hanwen Xing, Yiqi Xue, Steven Dillmann, and Han chung Lee. 2026. Skillsbench: Benchmarking how well agent skills work across diverse tasks.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, et al. 2023. Agentbench: Evaluating llms as agents. In *The Twelfth International Conference on Learning Representations*.

MiniMax. 2026. MiniMax M2.7: Early echoes of self-evolution. https://www.minimax.io/news/minimax-m27-en.

Moonshot AI. 2025. Kimi code CLI. https://github.com/MoonshotAI/kimi-cli.

OpenAI. 2025. Codex by openai | ai coding agent. https://openai.com/codex/.

OpenAI. 2026a. GPT-5.3-Codex system card. https://openai.com/index/gpt-5-3-codex-system-card/.

OpenAI. 2026b. GPT-5.4 thinking system card. Technical report, OpenAI.

PinchBench Contributors. 2026. PinchBench: Real-world benchmarks for AI coding agents. https://github.com/pinchbench/skill. GitHub repository.

Ivan Seleznov. 2026. Why claude code skills don't activate and how to fix it. Medium blog post.

Gowthami Somepalli, Anubhav Gupta, Kamal Gupta, Shramay Palta, Micah Goldblum, Jonas Geiping, Abhinav Shrivastava, and Tom Goldstein. 2024. Measuring style similarity in diffusion models. *arXiv preprint arXiv:2404.01292*.

Wei Tang, Yixin Cao, Yang Deng, Jiahao Ying, Bo Wang, Yizhe Yang, Yuyue Zhao, Qi Zhang, Xuanjing Huang, Yu-Gang Jiang, and Yong Liao. 2025. EvoWiki: Evaluating LLMs on evolving knowledge. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 948–964, Vienna, Austria. Association for Computational Linguistics.

Kimi Team, Yifan Bai, Yiping Bao, Guanduo Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, Zhuofu Chen, Jialei Cui, Hao Ding, Mengnan Dong, Angang Du, Chenzhuang Du, Dikang Du, Yulun Du, Yu Fan, Yichen Feng, Kelin Fu, Bofei Gao, Hongcheng Gao, Peizhong Gao, Tong Gao, Xinran Gu, Longyu Guan, Haiqing Guo, Jianhang Guo, Hao Hu, Xiaoru Hao, Tianhong He, Weiran He, Wenyang He, Chao Hong, Yangyang Hu, Zhenxing Hu, Weixiao Huang, Zhiqi Huang, Zihao Huang, Tao Jiang, Zhejun Jiang, Xinyi Jin, Yongsheng Kang, Guokun Lai, Cheng Li, Fang Li, Haoyang Li, Ming Li, Wentao Li, Yanhao Li, Yiwei Li, Zhaowei Li, Zheming Li, Hongzhan Lin, Xiaohan Lin, Zongyu Lin, Chengyin Liu, Chenyu Liu, Hongzhang Liu, Jingyuan Liu, Junqi Liu, Liang Liu, Shaowei Liu, T. Y. Liu, Tianwei Liu, Weizhou Liu, Yangyang Liu, Yibo Liu, Yiping Liu, Yue Liu, Zhengying Liu, Enzhe Lu, Lijun Lu, Shengling Ma, Xinyu Ma, Yingwei Ma, Shaoguang Mao, Jie Mei, Xin Men, Yibo Miao, Siyuan Pan, Yebo Peng, Ruoyu Qin, Bowen Qu, Zeyu Shang, Lidong Shi, Shengyuan Shi, Feifan Song, Jianlin Su, Zhengyuan Su, Xinjie Sun, Flood Sung, Heyi Tang, Jiawen Tao, Qifeng Teng, Chensi Wang, Dinglu Wang, Feng Wang, Haiming Wang, Jianzhou Wang, Jiaxing Wang, Jinhong Wang, Shengjie Wang, Shuyi Wang, Yao Wang, Yejie Wang, Yiqin Wang, Yuxin Wang, Yuzhi Wang, Zhaoji Wang, Zhengtao Wang, Zhexu Wang, Chu Wei, Qianqian Wei, Wenhao Wu, Xingzhe Wu, Yuxin Wu, Chenjun Xiao, Xiaotong Xie, Weimin Xiong, Boyu Xu, Jing Xu, Jinjing Xu, L. H. Xu, Lin Xu, Suting Xu, Weixin Xu, Xinran Xu, Yangchuan Xu, Ziyao Xu, Junjie Yan, Yuzi Yan, Xiaofei Yang, Ying Yang, Zhen Yang, Zhilin Yang, Zonghan Yang, Haotian Yao, Xingcheng Yao, Wenjie Ye, Zhuorui Ye, Bohong Yin, Longhui Yu, Enming Yuan, Hongbang Yuan, Mengjie Yuan, Haobing Zhan, Dehao Zhang, Hao Zhang, Wanlu Zhang, Xiaobin Zhang, Yangkun Zhang, Yizhi Zhang, Yongting Zhang, Yu Zhang, Yutao Zhang, Yutong Zhang, Zheng Zhang, Haotian Zhao, Yikai Zhao, Huabin Zheng, Shaojie Zheng, Jianren Zhou, Xinyu Zhou, Zaida Zhou, Zhen Zhu, Weiyu Zhuang, and Xinxing Zu. 2025. Kimi k2: Open agentic intelligence.

Colin White, Samuel Dooley, Manley Roberts, Arka Pal, Benjamin Feuer, Siddhartha Jain, Ravid Shwartz-Ziv, Neel Jain, Khalid Saifullah, Sreemanti Dey, Shubh-Agrawal, Sandeep Singh Sandha, Siddartha Venkat Naidu, Chinmay Hegde, Yann LeCun, Tom Goldstein, Willie Neiswanger, and Micah Goldblum. 2025. Livebench: A challenging, contamination-free LLM benchmark. In *The Thirteenth International Conference on Learning Representations*.

Renjun Xu and Yang Yan. 2026. Agent skills for large language models: Architecture, acquisition, security, and the path forward.

Yutao Yang, Junsong Li, Qianjun Pan, Bihao Zhan, Yuxuan Cai, Lin Du, Jie Zhou, Kai Chen, Qin Chen, Xin Li, Bo Zhang, and Liang He. 2026. Autoskill: Experience-driven lifelong learning via skill self-evolution.

Jiahao Ying, Yixin Cao, Yushi Bai, Qianru Sun, Bo Wang, Wei Tang, Zhaojun Ding, Yizhe Yang, Xuanjing Huang, and Shuicheng Yan. 2024a. Automating dataset updates towards reliable and timely evaluation of large language models. In *Advances in Neural Information Processing Systems*, volume 37, pages 17106–17132. Curran Associates, Inc.

Jiahao Ying, Yixin Cao, Kai Xiong, Long Cui, Yidong He, and Yongbin Liu. 2024b. Intuitive or dependent? investigating LLMs' behavior style to conflicting prompts. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4221–4246, Bangkok, Thailand. Association for Computational Linguistics.

Boyuan Zheng, Michael Y. Fatemi, Xiaolong Jin, Zora Zhiruo Wang, Apurva Gandhi, Yueqi Song, Yu Gu, Jayanth Srinivasa, Gaowen Liu, Graham Neubig, and Yu Su. 2025a. Skillweaver: Web agents can self-improve by discovering and honing skills.

Hao Zheng, Xinyan Guan, Hao Kong, Wenkai Zhang, Jia Zheng, Weixiang Zhou, Hongyu Lin, Yaojie Lu, Xianpei Han, and Le Sun. 2025b. Pptagent: Generating and evaluating presentations beyond text-to-slides. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 14413–14429.

Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, et al. 2023. Webarena: A realistic web environment for building autonomous agents. In *The Twelfth International Conference on Learning Representations*.

A Technical Appendices and Supplementary Material

A.1 Model Selection

Based on the proposed OpenSkillEval framework, we conduct a systematic evaluation of a range of state-of-the-art agent systems together with their corresponding foundation models. Our evaluation includes Claude Code (Anthropic, 2025a) with the Claude 4.6 series (Anthropic, 2026), Codex (OpenAI, 2025) with the GPT series (OpenAI, 2026a), Gemini CLI (Google, 2025) with the Gemini 3.1 Pro (Google DeepMind, 2026), Kimi Code CLI (Moonshot AI, 2025) with the Kimi K2.6 (Team et al., 2025) series, as well as adapted Minimax models (MiniMax, 2026) and DeepSeek V4 Pro (DeepSeek-AI, 2026) integrated into the Claude Code framework.

A.2 Experimental Environment

We adopt Harbor (Harbor Framework Team, 2026) as the unified execution framework for all experiments, and thank its open-source maintainers for their support and continued development. At runtime, all agents are executed under the same containerized environment based on `ubuntu:24.04`. To ensure fairness in environment setup and execution, we also provide network access during runtime and use unified timeout settings, with `build_timeout_sec = 1800.0 * 5` and `timeout_sec = 900.0 * 5`. All agents are evaluated using the official Responses API service of their corresponding model providers. Unless otherwise specified, all agent frameworks are run with their default parameter settings.[1]

For each instance, we provide the agent with the natural-language instruction I (stored as `Instruction.md`), the structured task specification T (stored as `task_input.json`), and any associated source materials or data files required by the task. These additional inputs vary by task category. For example, presentation generation is accompanied by `source_brief.md`, which may include tables, figures, and other supporting content; report generation is provided with the corresponding tabular dataset (e.g., `data.csv`); and data visualization is paired with a structured source file such as `source_data.json`. For each task category, we standardize the expected output format to ensure fair and consistent evaluation across agent systems. Specifically, the required output format is `.pptx` for presentation generation, a website with `index.html` as the entry point for front-end web design, `.png` for poster generation, `.html` for report generation, and `.png` for data visualization.

For evaluation, we primarily use Claude Opus 4.6 because of its strong capability, while also using Gemini 3.1 Pro for ablation analysis in Appendix A.5. For the agent-as-judge setting, we deploy Claude Opus 4.6 within the Claude Code framework and conduct evaluation in a Docker-based environment.

A.3 Task-Specific Evaluation

For VLM-based evaluation, we design task-specific input representations and evaluation granularity according to the characteristics of each task. The detailed matrices is shown in Table 3.

Presentation Generation. For presentation generation, we first convert the generated `.pptx` files into PDF format using Microsoft PowerPoint (version 16.108.1). We choose this conversion pipeline because we found that commonly used Linux-friendly conversion tools often introduce substantial rendering errors. The resulting PDF is then rasterized page by page into PNG images for per-slide evaluation of *content quality* and *visual design*. For *completeness* and *fidelity*, we provide the evaluator with the full set of slide screenshots, together with the task specification JSON and `source_brief.md` (including inline figures), so that it can make a global judgment based on both the output artifact and the source requirements.

Front-end Web Design. For front-end web design, we first use Claude 4.6 Opus to simulate user interaction with the generated website through vision- and DOM-grounded browsing. This step is necessary because web evaluation often involves multiple pages, navigation paths, and interaction states that cannot be captured by static screenshot. The browsing agent produces both screenshots and a structured browsing evaluation report containing page-level loading status, discovered sections, missing sections, and navigation outcomes. Based on these collected screenshots, we evaluate *visual design* and *responsiveness*. For responsiveness, we render the website under multiple viewport settings, including device sizes corresponding to iPhone 13 and iPad 7, to assess layout adaptation across screen sizes. In practice, we find

[1] For runtime stability and reproducibility, we fix the installed versions of the CLI-based agent frameworks in the execution environment.

that using only full-page screenshots often leads to excessive image compression and loss of fine-grained details. We therefore adopt a hybrid screenshot strategy: the evaluator receives both full-page screenshots for global structural understanding and viewport-based sequential screenshots (with a viewport height of 900px) for local detail inspection.

Poster Generation and Data Visualization. For poster generation and data visualization, the final artifacts are single PNG images, so we directly use the generated images as inputs to the VLM evaluator. The evaluation is grounded in the predefined task specification, including criteria such as *coverage of required sections* and *goal insight satisfaction*, in order to assess whether the produced artifact fulfills the requested content and communicative goals.

In addition, for data visualization, we further incorporate an agent-based evaluation component to assess *data accuracy*. Specifically, we provide the evaluator with both the execution trajectory of the task and the original plotting data file (e.g., `source_data.json`), and ask it to extract the chart-construction steps and verify whether the generated visualization correctly uses the intended data. This produces a step-level evaluation report focused on the correctness of data usage.

Report Generation. For report generation, we first convert the generated HTML report into PDF format, and then apply the same VLM-based screenshot evaluation strategy used for web design: we provide both full-page renderings and segmented screenshots to balance global structure assessment with fine-grained content inspection. For numerical claims and factual consistency, we further introduce an agent-based evaluation procedure. Specifically, we provide the evaluator with the generated HTML report together with the underlying Kaggle dataset, and ask it to analyze the reported claims step by step through code-based verification, producing a step-level evaluation report focused on data consistency and claim correctness.

A.4 Task Input Schemas

This section presents the task input schemas used in our benchmark construction pipeline. During automatic case generation, each collected source is transformed into a task-specific structured specification (stored as `task_input.json`) following the corresponding schema. These schemas define the core information required for each downstream task, and serve as the intermediate representation from source materials to executable task instances.

1. Data Visualization

```
{
  // Meta -- required
  "application": "data-visualization",
  "case_id":     "case-climate-trends",
  "language":    "en",

  // Style -- optional (omit to test agent
     autonomy)
  "style": {
    "theme":    "scientific",
    "audience": "researchers and policy
    makers",
    "tone":     "clean, publication-ready"
  },

  // Goal -- required (one insight per
     case; chart_type chosen by agent)
  "goal": {
    "id":      "warming-trend",
    "insight": "Show the accelerating
    global warming trend over 75 years"
  }
}
```

2. Poster Generation

```
{
  // Meta -- required
  "application": "poster-generation",
  "case_id":     "case-01-data-report",
  "language":    "en",

  // Poster constraints -- optional
  "poster": {
    "aspect_ratio": "landscape",   //
    landscape | portrait | square |
    A0-landscape | ...
    "audience":     "data-report",
    "tone":         "data-forward,
    professional",
  },

  // Content brief -- optional
  "brief": {
    "title":     "2025 AI Agent Trends",
    "one_liner": "One-line overview",
    "goal":      "What the audience should
    take away"
  },

  // Sections -- optional (agent plans
     structure if omitted)
  "sections": [
    { "id": "overview", "title": "Market
    Overview",
      "objective": "Show overall market
    size and growth" }
  ],
```

Task	Dimension	Method	Description
Presentation Generation	content_quality	VLM-judge	Per-slide text quality
	visual_design	VLM-judge	Per-slide visual aesthetics
	completeness	VLM-judge	Whole-deck task requirement coverage
	fidelity	VLM-judge	Whole-deck factual consistency with source_brief.md
Front-end Web Design	visual_design	VLM-judge	Aesthetics on full-page screenshots
	responsive	VLM-judge	Layout consistency across desktop / tablet / mobile
	navigation_pass_rate	Agent	Playwright tests inter-page navigation links
	interaction_pass_rate	Agent	Playwright tests interactive components (accordions, modals, etc.)
	data_display_pass_rate	Agent	Playwright extracts displayed content vs. expected items
Poster Generation	content_quality	VLM-judge	Data accuracy and traceability to source_brief.md
	visual_design	VLM-judge	Color, layout, typography, polish
	completeness	VLM-judge	Coverage of required sections / metrics
Data Visualization	insight_expression	VLM-judge	Whether the chart conveys the goal insight
	visual_quality	VLM-judge	Color, layout, label completeness
	completeness	VLM-judge	Task requirement fulfillment
	data_accuracy	Agent	Traces trajectory.json to verify data lineage to source_data.json
Report Generation	content_quality	VLM-judge	Writing quality, clarity, depth of analysis
	visual_quality	VLM-judge	Chart selection, color, labels, readability
	completeness	VLM-judge	Coverage of required sections / KPIs
	data_accuracy	Agent	Python code compares numbers in report vs. data.csv
	fidelity	Agent	Verifies extracted claims against data.csv

Table 3: Evaluation matrix across the five OpenSkillEval task categories. Each task is decomposed into multiple evaluation dimensions, which are scored either by a VLM judge (using visual or textual rubrics on a 1–5 scale) or by an evaluation agent (using programmatic verification to produce a score or pass rate). To enable consistent comparison across metrics, pass rates are linearly mapped to the 1–5 scale via $4x + 1$.

```
  // Metrics -- optional
  "metrics": [
    { "name": "Market Size", "current":
     "$12B", "target": "$45B (2027)" }
  ]
}
```

3. Presentation Generation

```
{
  // Meta -- required
  "application": "ppt-generation",
  "case_id":     "case-01-internal-review",
  "language":    "en",

  // Deck constraints -- optional
  "deck": {
    "aspect_ratio": "16:9",
     // default 16:9
    "slide_count":  6,
      // omit to let agent decide
    "audience":     "internal product
    review",
    "tone":         "professional, concise"
  },

  // Content brief -- optional
  "brief": {
    "title":     "SkillArena Q4 Proposal",
    "one_liner": "One-line overview",
    "goal":      "What the audience should
    understand or decide"
  },

  // Slide blueprint -- optional (agent
    plans deck if omitted)
  "slides": [
    { "id": "why-now", "title": "Why Now",
      "objective": "Help reviewers
    understand the launch context" }
    // ...
  ]
}
```

4. Report Generation

```
{
  // Meta -- required
  "application": "report-generation",
  "case_id":     "case-01-sales-analysis",
  "language":    "en",

  // Report constraints -- optional
  "report": {
    "type":     "sales-report",
    "audience": "management",
    "tone":     "professional, data-forward"
  },

  // Content brief -- optional
  "brief": {
    "title":     "2024 Q4 Sales Performance
    Report",
    "one_liner": "One-line overview",
    "goal":      "What readers should take
    away"
  },
```

```
  // Section blueprint -- optional
  "required_sections": [
    { "section_id": "executive-summary",
      "title":      "Executive Summary",
      "objective":  "Key findings and core
    metrics at a glance" }
    // ...
  ],

  // KPI definitions -- optional
  "kpis": [
    { "name": "Total Sales", "description":
    "All Q4 sales revenue" }
  ],

  // Analysis dimensions -- optional
  "analysis_dimensions": ["product",
    "region", "time_trend"]
}
```

5. Web Design

```
{
  // Meta -- required
  "application": "web-design",
  "case_id":     "case-01-landing-page",
  "language":    "en",

  // Site constraints -- optional
  "site": {
    "type":       "landing-page",
    "page_count": 2,
     // omit to let agent decide
    "audience":   "developers and technical
    decision-makers",
    "tone":       "modern, professional,
    bold",
    "responsive": true,
      // default true
    "dark_mode":  false
      // default false
  },

  // Content brief -- optional
  "brief": {
    "title":     "TechPulse -- API
    Platform",
    "one_liner": "Next-gen API management
    and deployment",
    "goal":      "Help visitors grasp the
    value prop and sign up"
  },

  // Page blueprint -- optional
  "pages": [
    { "page_id": "home", "file":
    "index.html",
      "sections":  ["hero", "features",
    "testimonials", "cta", "footer"],
      "objective": "Homepage with value
    prop and sign-up CTA" },
    { "page_id": "pricing", "file":
    "pricing.html",
      "sections":  ["pricing-tiers", "faq",
    "cta", "footer"],
      "objective": "Pricing page with
    three-tier comparison" }
  ],

  // Cross-page navigation -- optional
  "navigation": [
    { "from": "home", "trigger": "navbar
    'Pricing' link", "to": "pricing" },
    { "from": "home", "trigger": "'Get
    Started' CTA button", "to": "pricing" }
  ],

  // In-page interactions -- optional
  "interactions": [
    { "id": "faq-accordion", "page": "home",
      "trigger":  "click on FAQ question",
      "expected": "expand to show the
    answer; click again to collapse",
      "type":     "toggle" },
    { "id": "pricing-switch", "page":
    "pricing",
      "trigger":  "toggle 'Monthly /
    Yearly' switch",
      "expected": "all three tier prices
    update in sync",
      "type":     "state-change" }
  ],

  // Required data display -- optional
  "data_display": [
    { "id": "pricing-table", "page":
    "pricing", "type": "table",
      "source_ref":       "comparison table
    in source_brief.md",
      "expected_content": ["Free", "Pro",
    "Enterprise", "10K/mo", "1M/mo"],
      "description":      "Feature
    comparison across three pricing tiers" }
    ,
    { "id": "hero-stats", "page": "home",
    "type": "stats",
      "source_ref":       "core metrics in
    source_brief.md",
      "expected_content": ["50,000+",
    "99.99%", "50ms"],
      "description":      "Show 3-4 key
    numerical metrics in the hero area" }
  ]
}
```

A.5 Human Evaluation

Although our framework is fully automatic and can continuously generate new test cases and conduct evaluation at scale, we further perform human evaluation to assess both the quality of the generated task instances and the reliability of the automatic judgments. Specifically, we involve four senior researchers in natural language processing as human annotators, and randomly sample 100 task instances balanced across the five task categories. For task-instance quality, annotators score each gener-

ated case on three dimensions: *fluency*, *coherence*, and *completeness*, where completeness measures whether the task description is sufficiently clear and well specified for execution. Each dimension is rated on a 1–3 scale. Across all dimensions, the mean human score exceeds 2.98, with an exact agreement rate of 98.8%. Additional details are provided in Appendix A.5. For evaluation quality, given the complexity of the overall evaluation pipeline, we perform human assessment directly on the automatic evaluation results. Annotators are asked to judge whether the assigned score and its accompanying rationale are consistent with the task requirements and the generated output. We use a 1-3 scale: a score of 3 indicates that both the judgment and the rationale are reasonable and well aligned with the task requirements; a score of 2 indicates that the assigned score is reasonable but the rationale is incomplete or not convincing; and a score of 1 indicates that the assigned score itself is unreasonable. Across four annotators, the pooled mean rating is 2.86. Exact four-way agreement accounts for 75.0% of annotated units, and averagely only 6.01% of cases receive a rating of 1. These results suggest that the proposed automatic evaluation pipeline is reasonably reliable. In addition, we use another visually strong model from a different family, Gemini 3.1 Pro, as an auxiliary evaluator to further examine the robustness of our automatic evaluation pipeline. On a held-out set of 200 sampled cases, the scores produced by Gemini 3.1 Pro show strong correlation with our primary evaluator, achieving Pearson and Spearman correlations of 0.855 and 0.821, respectively.

To validate both the quality of the automatically generated task instances and the reliability of our Automatic Evaluation Pipeline, we conduct a human evaluation study. Specifically, we randomly sample 100 task instances, balanced across the five task categories, and ask four senior researchers in natural language processing to perform the assessment. For each sampled instance, evaluators are provided with: (1) the task input, including the natural-language instruction and corresponding task specification; and (2) the generated output artifact. Because many of the task instances involve complex files and multimodal outputs, we build a dedicated web-based interface to support side-by-side inspection. As shown in Figure 6, the interface presents the task details and reference materials in a structured format for easier reading. We also provide task-specific artifact visualizations, as shown in Figure 7. For example, web-design outputs are deployed in an interactive front-end environment for direct inspection, while generated slide decks are converted into PDFs for convenient review.

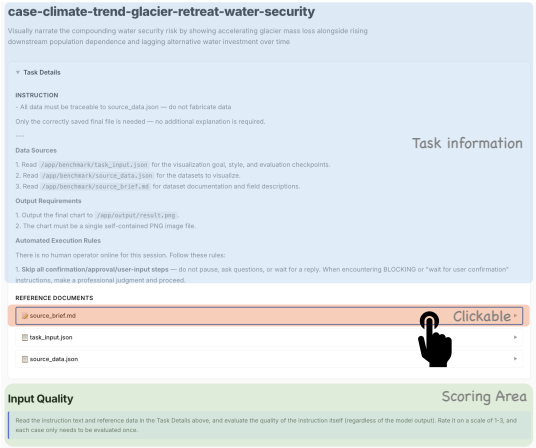


Figure 6: Web-based interface for human evaluation of generated task instances.

The detailed guidelines for evaluating the generated task instances are shown below. For artifact evaluation, we directly adopt the evaluation prompts used in our VLM-based automatic evaluation pipeline; full details are provided in Appendix A.8.

Task Instances Evaluation Guideline

Read the instruction text and reference data in the task details above, and evaluate the quality of the instruction itself (regardless of the model output). Rate it on a scale of 1–3. Each case only needs to be evaluated once.

Fluency

3 Grammatically correct, fluent and natural, accurate terminology
2 Minor grammar or wording issues, slightly affects readability
1 Multiple grammar errors or unclear wording, severely impedes comprehension

Coherence

3 Clear goal, logically consistent requirements; expected output is unambiguous after reading
2 Goal is mostly clear, but some requirements are vague or ambiguous
1 Description is chaotic, requirements conflict or are severely vague

Completeness

3 All essential information present: data source specified, output format clear, constraints stated
2 Core information present, but some details are missing; agent must make assumptions
1 Critical information is missing; agent cannot reliably complete the task

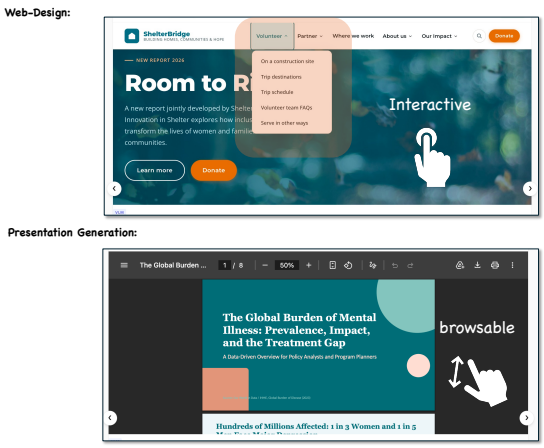


Figure 7: Artifact inspection interface used in human evaluation. The system provides task-specific visualization of generated outputs, including interactive web pages for front-end design tasks and converted PDF views for presentation outputs.

A.6 Case and Skills Statistic

We summarize the constructed task instances and collected skills in this section. Figure 8 shows the distribution of generated cases across the five downstream task categories. Table 4 reports the selected skills for each category. In total, we collect 30 community-contributed skills, with 4-8 skills per task, enabling controlled comparison of skill effects within each downstream setting. Since open-source skill repositories are continuously updated, these statistics should be understood as a snapshot of the ecosystem at the time of collection.

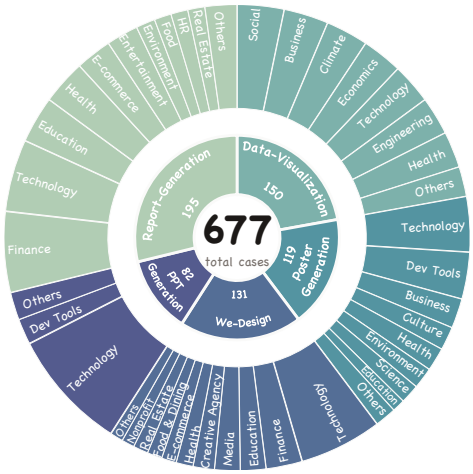


Figure 8: Case statistics

Task	#	Skills
Presentation Generation	6	anthropics-pptx, frontend-slides, minimax-pptx, powerpoint-pptx, ppt-master, pptx-manipulation
Front-end Web Design	8	frontend-design-ultimate, web-design-expert, loom, seo-local-business, SuperDesign, ui-styling, ui-ux-pro-max, web-frameworks
Poster Generation	4	antv-infographic, paper-poster, canvas-design, visualize
Data Visualization	6	anthropics, mermaid-tools, mermaidjs, tufte, visual-explainer, visualize
Report Generation	6	claude-office, chatgpt-pdf, excel-report, business-auto, clawfu, devkit

Table 4: Skills selected for each task (30 total). Names are clickable. Because these repositories are continuously updated by the community, the collected skill set is time-sensitive and should be understood as a snapshot of the open skill ecosystem at the time of collection.

A.7 More Experimental Result

We provide additional results for skill performance, cost, and stylistic effects. Figure 9 shows the full performance-cost trade-off across all model-task pairs, highlighting the heterogeneous impact of skill augmentation. Figures 10a, 10b, and 11 report diversity changes induced by skills in presentation, poster, and web-design tasks. Table 5 further summarizes the assets, constraints, and observed outcomes of visual-generation skills, linking skill design to quality and diversity.

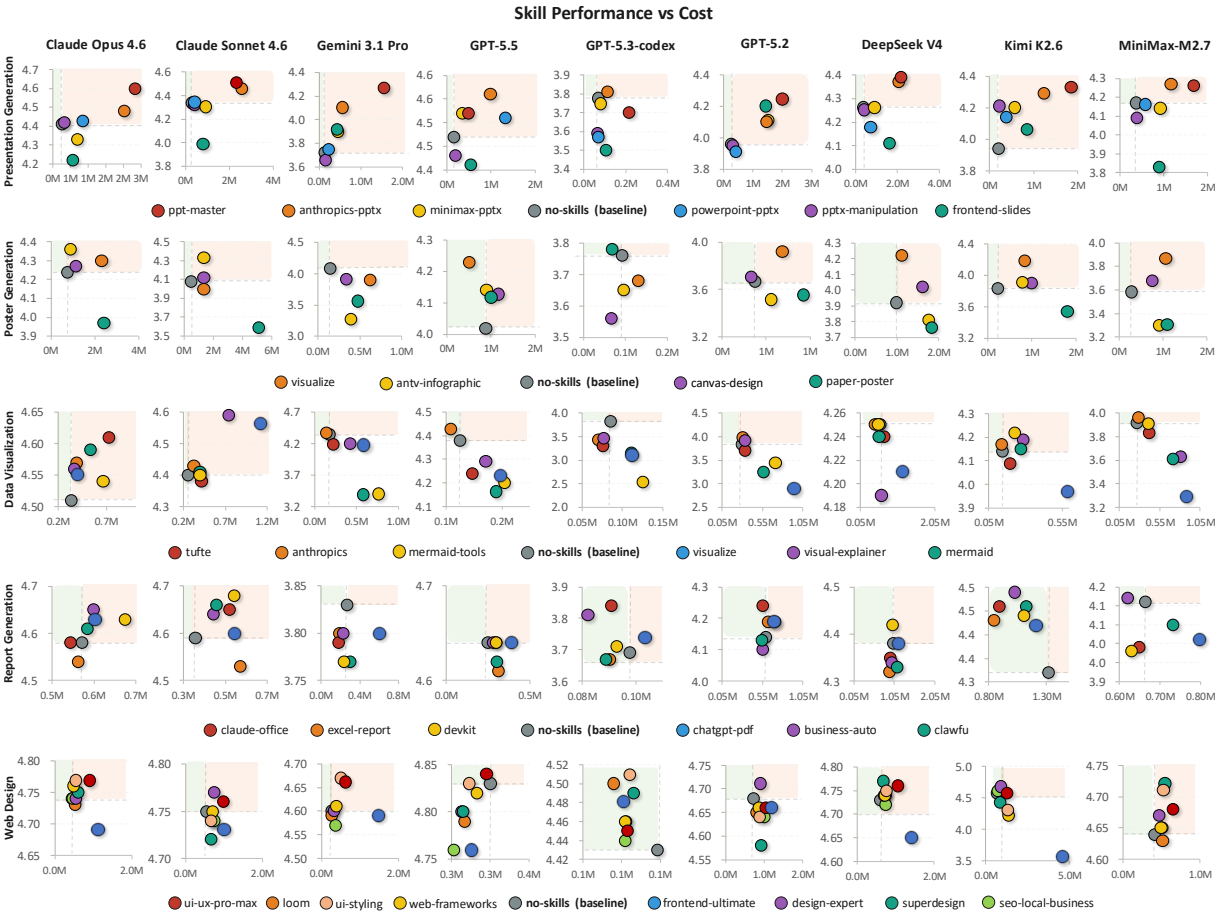
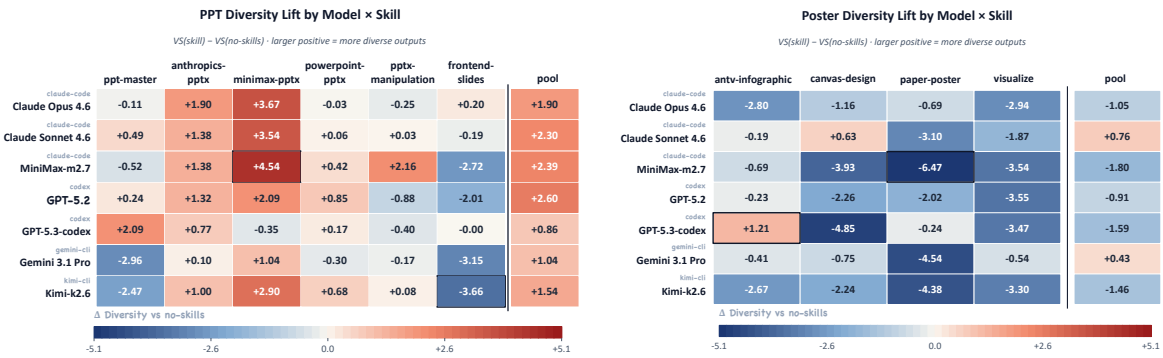


Figure 9: Skill performance versus cost across tasks and agent systems. Each subplot corresponds to one model-task pair, where the x-axis shows average token cost and the y-axis shows overall task performance. Colored points denote different skills, while the gray point marks the *no-skills* baseline. The dashed vertical and horizontal lines indicate the baseline cost and performance, respectively, so that points in the upper-left region represent the most desirable outcomes: higher quality at lower cost. The results show that skill augmentation is highly heterogeneous across models and tasks: some skills consistently improve performance, while others increase cost without yielding meaningful gains.



(a) (b)

Figure 10: Impact of skills on stylistic diversity relative to the *no-skills* baseline, measured by changes in within-group Vendi Score computed from CSD-ViT-L style embeddings. **(a)** Presentation generation. **(b)** Poster generation. The *pool* column aggregates outputs across all skills for cross-skill analysis.

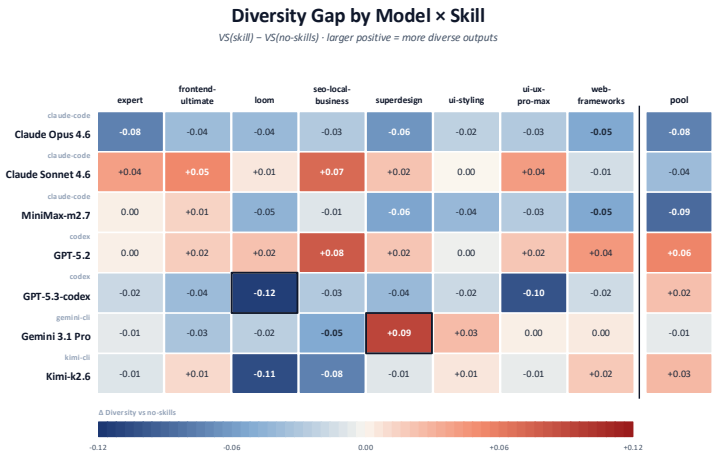


Figure 11: Impact of web design skills on stylistic diversity relative to the *no-skills* baseline, measured by changes in within-group Vendi Score computed from CSD-ViT-L style embeddings. Positive values indicate more diverse outputs under a given skill, while negative values indicate stronger stylistic convergence. The *pool* column aggregates outputs across all skills for cross-skill analysis.

Skill	Key Assets	Key Constraints	A	C	Obs. O	Obs. D	Obs. E	Obs. Div
Presentation Generation *ppt-generation*								
ppt-master	101 layouts + 33 charts + 640 icons + 39 references	25 MUST + 10 NEVER	5	5	4.27±	3.91±	4.42±	-0.46
anthropics-pptx	2 references (pptxgenjs, editing) + 78 inline design lines	1 MUST + 3 NEVER	3	2	4.20±	3.65±	4.46±	+1.12
minimax-pptx	5 references (incl. design system) + 8 inline design lines	9 MUST + 1 NEVER	2	2	4.09±	3.47±	4.40±	+2.49
powerpoint-pptx	3 references (slides, charts, design) + 21 inline design lines	2 MUST	1	1	4.03±	3.25±	4.47±	+0.27
frontend-slides	Viewport-base CSS + 12-style preset library	13 MUST + 7 NEVER	2	5	3.96±	3.52±	4.24±	-1.65
pptx-manipulation	Unpack/repack tutorial + 16 inline design lines	—	1	0	4.02±	3.29±	4.43±	+0.08
no-skills	—	—	0	0	4.04±	3.20±	4.54±	—
Poster Generation *poster-generation*								
visualize	Mandatory skeleton + 9 references + 98 inline design lines	117 MUST + 28 NEVER	4	5	4.05±	3.93±	4.11±	-2.75
canvas-design	27 inline design lines	15 MUST + 5 NEVER	1	5	3.91±	3.56±	4.09±	-2.08
antv-infographic	External AntV template library	DSL grammar	1	2	3.77±	3.47±	3.92±	-0.83
paper-poster	External LaTeX tcbposter class + 256 inline design lines	21 MUST + 15 NEVER	2	5	3.67±	2.88±	4.07±	-3.07
no-skills	—	—	0	0	3.95±	3.56±	4.14±	—
Front-end Web Design *web-design*								
ui-ux-pro-max	16 frontend-stack guides + 14 design CSVs	44 MUST + 10 NEVER	5	5	4.67±	4.41±	4.88±	-0.01
ui-styling	7 references + 2 generator scripts + 321 inline lines	—	3	0	4.66±	4.39±	4.88±	-0.01
web-design-expert	3 references + 220 inline lines	1 NEVER	2	1	4.66±	4.35±	4.90±	-0.01
loom	1 UX-review reference + motivation/personality prompts	1 MUST	1	1	4.65±	4.34±	4.90±	-0.01
seo-local-business	3 SEO templates (head, robots, sitemap) + 332 inline lines	4 MUST	2	2	4.64±	4.30±	4.89±	-0.00
web-frameworks	8 references + 2 init scripts	—	3	0	4.63±	4.31±	4.88±	-0.01
superdesign	212 inline design lines, no external references	2 NEVER	1	1	4.55±	4.19±	4.84±	+0.02
frontend-ultimate	3 references + site-config template + 408 inline lines	1 MUST + 1 NEVER	2	1	4.54±	4.24±	4.80±	+0.01
no-skills	—	—	0	0	4.65±	4.32±	4.91±	—

Table 5: Quantifying presentation, poster, and web-design skills along two orthogonal dimensions and relating them to observed quality and stylistic diversity. **Asset score** (A, 0–5) measures the strength of *design priors* packaged with a skill, based on countable signals such as visual template files, reference design documents, design-data size, and design-related line count inside `SKILL.md`. **Constraint score** (C, 0–5) measures the strength of *procedural constraints*, based on MUST-style and NEVER-style directive counts together with structural locks. **Obs.** O denotes the measured overall judge score; **Obs.** D denotes the design sub-score, taken directly from the judge for presentation and poster generation and computed as the mean of (Visual Design + Responsive) for web design; **Obs.** E denotes the content-oriented effectiveness score, computed as the mean of (Completeness + Fidelity) for presentation generation, the mean of (Content + Completeness) for poster generation, and the mean of three pass-rate for web design; **Obs. Div** denotes the observed visual-diversity score.

A.8 Evaluation Prompt

We use the following rubrics for LLM-as-a-judge evaluation. Each rubric specifies the inputs provided to the judge and the scoring criteria.

1. Data Visualization

Insight Expression **single image**

```
Evaluate the **insight expression** of
    this data visualization.

The visualization was created to convey a
    specific insight:

**Goal insight**: {insight}

Criteria:
- Does the chosen visualization type
    effectively communicate this insight?
- Can the reader **actually** understand
    the key message at a glance, as
    rendered?
- Are there effective annotations,
    highlights, or emphasis that draw
    attention to the insight?
- Is the presentation approach well-suited
    to the data characteristics?
- Does the visualization tell a clear
    story, or does the reader have to work
    to extract meaning?

Important: Judge the **end result as the
    reader would experience it**. A
    structurally sound approach that is
    undermined by severe rendering
    problems (e.g., overlapping labels
    hiding key annotations, illegible text
    in critical areas) should be scored
    lower -- the insight is only
    "expressed" if the reader can actually
    perceive it. Conversely, do NOT
    double-penalize minor cosmetic issues
    already covered by Visual Quality;
    only penalize rendering problems that
    **directly obstruct** the target
    insight.

Note: The creator was free to choose ANY
    chart type or visual approach. Do NOT
    penalize for unconventional choices --
    judge ONLY whether the insight is
    effectively conveyed.

Scoring (1-5):
- 5: Reader grasps the key message
    immediately as rendered, clever use of
    annotation/emphasis/contrast, all
    insight components clearly visible
- 4: Insight clearly expressed, reasonable
    approach, minor gaps in emphasis or
    supporting elements
- 3: Insight basically understandable but
    presentation is generic or partially
    obscured, reader must interpret on
    their own
- 2: Presentation approach mismatches the
    insight goal, or key information is
    buried/illegible
- 1: Cannot read the target insight from
    the visualization

Return JSON: {"score": <1-5>, "reason":
    "<brief explanation>"}
```

Data Accuracy **single image + source text**

```
Evaluate the **data accuracy and factual
    fidelity** of this data visualization.

You are given the visualization, the
    original data, and background context.
    Check:
- Are the data points, proportions, and
    trends consistent with the source data?
- Are axis scales, labels, and values
    correct?
- Are any annotations or callouts
    traceable to the source material?
- Is there any fabricated data or
    misleading representation?

--- SOURCE DATA ---
{source_data}
--- END SOURCE DATA ---

--- SOURCE CONTEXT ---
{source_brief}
--- END SOURCE CONTEXT ---

Scoring (1-5):
- 5: All data points, proportions, and
    trends match source exactly;
    annotations traceable to source; no
    fabrication
- 4: Core data correct, minor deviations
    in secondary data points or slight
    extrapolation
- 3: Mostly correct, but notable numerical
    errors, proportion distortion, or
    untraceable annotations
- 2: Multiple data inconsistencies with
    source, or obvious fabricated content
- 1: Extensive data errors or fabrication,
    visualization is not trustworthy

Return JSON: {"score": <1-5>, "reason":
    "<brief explanation>"}
```

Visual Quality **single image**

```
Evaluate the **visual quality** of this
    data visualization.

Criteria:
- Color scheme: harmonious palette,
    appropriate for the topic,
    colorblind-friendly if applicable
- Layout: clean arrangement, proper
    spacing, nothing overlapping
- Labels and annotations: title, axis
    labels, legend, units -- all present
```

```
    and readable
- Typography: readable fonts, clear size
    hierarchy
- Professional finish: publication-ready
    quality, attention to detail

Scoring (1-5):
- 5: Harmonious colors, polished layout,
    complete labels
    (title/axes/legend/units),
    publication-grade professional quality
- 4: Good colors and layout, labels mostly
    complete, minor flaws (uneven spacing,
    label overlap)
- 3: Basic colors, functional layout but
    lacks design sophistication, some
    labels missing
- 2: Monotonous or clashing colors, rough
    layout, labels severely lacking
- 1: Chaotic styles, overlapping elements,
    visually unacceptable

Return JSON: {"score": <1-5>, "reason":
    "<brief explanation>"}
```

Completeness **single image**

```
Evaluate the **task completeness** of this
    data visualization.

The task requirements are provided below.
    Check whether the goal's insight is
    expressed and all requirements are met.

Task requirements:
{task_requirements}

Scoring (1-5):
- 5: Goal insight fully expressed, all
    requirements completely satisfied
- 4: Core requirements met, minor
    omissions (e.g., missing unit or a
    specific annotation)
- 3: Most requirements met, but notable
    gaps
- 2: Only partially satisfied
- 1: Key requirements not met, output
    unusable

Return JSON: {"score": <1-5>, "reason":
    "<brief explanation>"}
```

2. Poster Generation

Design **single image**

```
Evaluate the **visual design quality** of
    this poster/infographic.

Criteria:
- Color scheme: harmonious palette,
    appropriate for the topic and tone
- Layout: clean alignment, proper spacing,
    clear visual hierarchy
- Typography: readable fonts, clear size
    hierarchy (title > heading > body)
- Consistency: unified style throughout
    (colors, fonts, spacing)
- Professional polish: attention to
    detail, no overlapping elements

Scoring (1-5):
- 5: Harmonious colors, polished layout,
    clear hierarchy, professional design
    quality
- 4: Good colors and layout, minor flaws
    (slight misalignment, spacing issues)
- 3: Basic color scheme, layout functional
    but lacks design sophistication
- 2: Monotonous or clashing colors, rough
    layout
- 1: Chaotic styles, overlapping elements,
    hard to read

Return JSON: {"score": <1-5>, "reason":
    "<brief explanation>"}
```

Completeness **single image**

```
Evaluate the **task completeness** of this
    poster/infographic.

The task requirements are provided below.
    Check whether ALL requirements are met.

Task requirements:
{task_requirements}

Scoring (1-5):
- 5: All requirements fully satisfied
- 4: Core requirements met, minor omissions
- 3: Most requirements met, some notable
    gaps
- 2: Only partially satisfied
- 1: Key requirements not met

Return JSON: {"score": <1-5>, "reason":
    "<brief explanation>"}
```

3. PPT Generation

Content **per-slide image**

```
Evaluate the **content quality** of this
    presentation slide.

Judge how effectively this slide delivers
    its key message to the reader.

Criteria:
- Key message: does the slide have a clear
    takeaway that the reader can grasp?
- Information density: appropriate amount
    of content (not too crowded, not too
    sparse)
- Clarity: text is well-written,
    grammatically correct, easy to
    understand
```

```
- Text-visual balance: charts/images
    complement the text and reinforce the
    message

If rendering issues (truncated text, blank
    charts) cause information to be LOST,
    deduct proportionally to how much of
    the slide's message is affected. But
    if the core message still comes
    through despite minor visual flaws, do
    not over-penalize -- visual polish is
    scored under Design.

Scoring (1-5):
- 5: Clear key message, well-developed
    points, visuals and text complement
    each other
- 4: Key message clear, good content, but
    minor gaps (e.g., a chart lacks
    labels, text slightly sparse)
- 3: Core information present but message
    is diluted (e.g., missing title, no
    clear takeaway, or significant content
    lost to rendering)
- 2: Key message unclear, content poorly
    organized or mostly lost to rendering
    issues
- 1: No discernible message -- slide is
    blank, empty, or content entirely
    unreadable

Return JSON: {"score": <1-5>, "reason":
    "<brief explanation>"}
```

Design **per-slide image**

```
Evaluate the **visual design** of this
    presentation slide.

Criteria:
- Color scheme: harmonious, appropriate
    for the tone
- Layout: clean alignment, proper spacing,
    no overlapping elements
- Typography: readable fonts, clear
    hierarchy (title vs body)
- Visual elements: backgrounds, icons,
    shapes that enhance the message
- Consistency: matches the overall deck
    style

Scoring (1-5):
- 5: Harmonious colors, engaging visual
    elements, professional and polished
- 4: Good colors with some visual
    elements, minor design flaws
- 3: Basic color scheme, with rough layout
    and no supplementary visual elements
- 2: Monotonous black/white, readable but
    unappealing
- 1: Conflicting styles, content difficult
    to read

Return JSON: {"score": <1-5>, "reason":
    "<brief explanation>"}
```

Completeness **full deck images**

```
Evaluate the **task completeness** of this
    presentation.

Check whether the required CONTENT is
    present in the slides. For each
    requirement below, determine if the
    corresponding content exists in the
    presentation.

Task requirements:
{task_requirements}

Important:
- Judge whether the required content IS
    PRESENT, not how it looks visually.
- A chart that renders blank or empty
    counts as MISSING content (deduct).
- Text that is overlapping or hard to read
    but IS present counts as COMPLETE (do
    not deduct -- visual issues are scored
    under Design).

Scoring (1-5):
- 5: All required content present -- every
    section, data point, and chart
    accounted for
- 4: Core content present, minor omissions
    (e.g., one data point or detail
    missing)
- 3: Most content present, but some
    required sections or charts are missing
- 2: Only partially complete -- multiple
    required elements missing
- 1: Key required content not present

Return JSON: {"score": <1-5>, "reason":
    "<brief explanation>"}
```

4. Report Generation

Content Quality **report text only**

```
Evaluate the **content quality** of this
    report across two aspects: writing
    quality AND analysis depth.

A. Writing & Structure:
- Organization: clear headings, logical
    flow, well-structured executive summary
- Clarity: well-written, grammatically
    correct, easy to understand
- Information density: appropriate amount
    of content (not too sparse, not
    overwhelming)
- Logical consistency: no
    self-contradicting claims or
    misleading interpretations

B. Analysis Depth:
- Are insights substantive (backed by
    statistics specific data) or
    surface-level (just restating numbers)?
- Does the analysis go beyond descriptive
    statistics (e.g., correlation,
```

```
    attribution, comparison)?
- Are conclusions and recommendations
    data-driven and specific (not generic
    advice)?
- Is there unnecessary redundancy (e.g.,
    same data presented in multiple
    formats without adding value)?

Scoring (1-5):
- 5: Excellent writing AND deep analysis
    -- insights are substantive,
    data-driven, logically consistent,
    with no redundancy
- 4: Good writing and organization, but
    some sections lack analytical depth or
    have minor redundancy
- 3: Adequate structure but analysis is
    mostly surface-level, or notable
    redundancy/inconsistency
- 2: Poor organization, stiff writing, and
    shallow analysis
- 1: Chaotic structure, grammar errors, no
    real analysis

Return JSON: {"score": <1-5>, "reason":
    "<brief explanation>"}
```

Visualization **full-page overview + scrolled crops**

```
Evaluate the **visualization quality** of
    this report.

Criteria:
- Chart type selection: appropriate for
    the data (bar for comparison, line for
    trend, pie for composition, etc.)
- Color scheme: professional, harmonious,
    accessible
- Labels and annotations: axis titles,
    data labels, legends, units
- Readability: data is immediately clear
    from the visualization
- Tables: properly formatted, aligned,
    with clear headers

Scoring (1-5):
- 5: Chart types perfectly matched to
    data, professional colors, complete
    annotations, data immediately clear
- 4: Good chart types, has titles and
    labels, good visual effect
- 3: Basic charts present, but type
    selection or labels could be improved
- 2: Few charts or wrong types, missing
    labels
- 1: No charts or charts are
    incomprehensible

Return JSON: {"score": <1-5>, "reason":
    "<brief explanation>"}
```

Completeness **report text only**

```
Evaluate the **task completeness** of this
    report.

Go through EVERY requirement below and
    check whether it is substantively
    addressed in the report. Do NOT just
    check if a section heading exists --
    verify that the actual content within
    each section fulfills what was asked.
    For example, if a KPI is required
    "segmented by genre and year", check
    that BOTH segmentations are present,
    not just one.

Task requirements:
{task_requirements}

Scoring (1-5):
- 5: Every requirement substantively
    addressed -- all sections have the
    requested content, all KPIs present
    with required segmentation
- 4: Core requirements met, but 1-2 minor
    details missing (e.g., a KPI lacks one
    segmentation dimension)
- 3: Most requirements met, but notable
    gaps (e.g., a section is present but
    missing key requested analysis)
- 2: Only partially satisfied -- multiple
    sections lack required content
- 1: Key requirements not met

Return JSON: {"score": <1-5>, "reason":
    "<brief explanation>"}
```

5. Web Design

Visual Design **per-page multi-image (full + crops)**

```
Evaluate the **visual design execution
    quality** of this web page.

Criteria:
- Color & typography: harmonious palette,
    readable fonts, clear heading
    hierarchy (h1 > h2 > body), consistent
    font sizing
- Layout & structure: well-organized
    sections, clear information hierarchy,
    consistent grid alignment, no
    misaligned or jagged elements
- Spacing & polish: appropriate white
    space between sections, no elements
    overlapping or clipped, no exposed
    HTML tags or rendering artifacts
- Placeholder handling: the task does NOT
    provide image/video assets, so
    placeholder elements are expected. Do
    NOT penalize for the absence of real
    images. Instead, evaluate whether
    placeholders are well-designed --
    appropriate size and position,
    consistent styling with the overall
    theme (e.g., labeled gray boxes are
    better than empty black voids)
```

```
Note: the first image is a fullpage
    overview, subsequent images are detail
    crops scrolled top-to-bottom. Boundary
    cuts between detail images are
    screenshot artifacts.

Scoring (1-5):
- 5: Harmonious color palette with
    intentional accent choices; typography
    has clear h1/h2/body hierarchy with
    consistent sizing; sections are
    well-separated with balanced spacing;
    grid alignment is pixel-consistent;
    placeholders are styled to match the
    theme (e.g., colored boxes with
    labels, icon placeholders)
- 4: Color palette is coordinated,
    typography hierarchy is clear, layout
    is well-organized; has 1-2 minor
    issues such as: one section has
    slightly uneven spacing, a placeholder
    is slightly oversized, a font weight
    is inconsistent in one area
- 3: Has a color scheme and basic section
    structure, but execution is noticeably
    rough: spacing is uneven across
    sections, heading sizes are
    inconsistent, placeholders are
    unstyled empty blocks, or the page
    feels like a wireframe with colors
    applied
- 2: Multiple visible problems: elements
    overlap or get clipped, sections are
    misaligned, exposed HTML tags or
    broken CSS visible, color choices
    clash, or large portions of the page
    are visually broken
- 1: Page renders as unstyled HTML (no CSS
    applied), or the page fails to load /
    displays a blank screen

Return JSON: {"score": <1-5>, "reason":
    "<brief explanation>"}
```

Responsive **per-device screenshots (mobile / tablet)**

```
Evaluate the **responsiveness** of this
    website on a {device_name}.

You are shown screenshots captured with
    {device_description}.
Each screenshot is from a different page
    of the same website. Evaluate the
    overall responsive quality across all
    pages shown.

Criteria:
- No horizontal scrollbar / content
    overflow
- Navigation is accessible (hamburger menu
    or adapted nav)
- Touch targets are appropriately sized
    (>=44px)
- Text is readable without zooming
- Content adapts to the viewport width (no
    desktop layout forced into a smaller
    screen)

Note: the task does NOT provide
    image/video assets. Pages may contain
    elements explicitly labeled or styled
    as placeholders (e.g., boxes with
    "placeholder" text, icons indicating
    missing media). Do NOT penalize these
    as a responsive issue -- only evaluate
    how the layout, navigation, and
    content elements adapt to this
    viewport.

Scoring (1-5):
- 5: No overflow, navigation properly
    adapted (e.g., hamburger menu on
    mobile), touch targets well-sized,
    content reflows cleanly to viewport
    width
- 4: Good adaptation overall; 1-2 minor
    issues such as: nav items slightly
    tight, one button partially clipped at
    edge, or slight spacing inconsistency
- 3: Has viewport meta and attempts
    responsive layout, but noticeable
    problems: nav not collapsed, some text
    requires horizontal scroll, or several
    elements are cramped
- 2: Layout significantly broken: content
    overflows viewport, navigation is
    unusable, or desktop layout is forced
    into the smaller screen
- 1: No responsive handling at all -- page
    renders at desktop width requiring
    zoom and scroll

Return JSON: {"score": <1-5>, "reason":
    "<brief explanation>"}
```

A.9 Case Study

This section presents representative case studies across the five downstream task categories in OpenSkillEval. Each example illustrates one generated task instance and its corresponding artifact output.

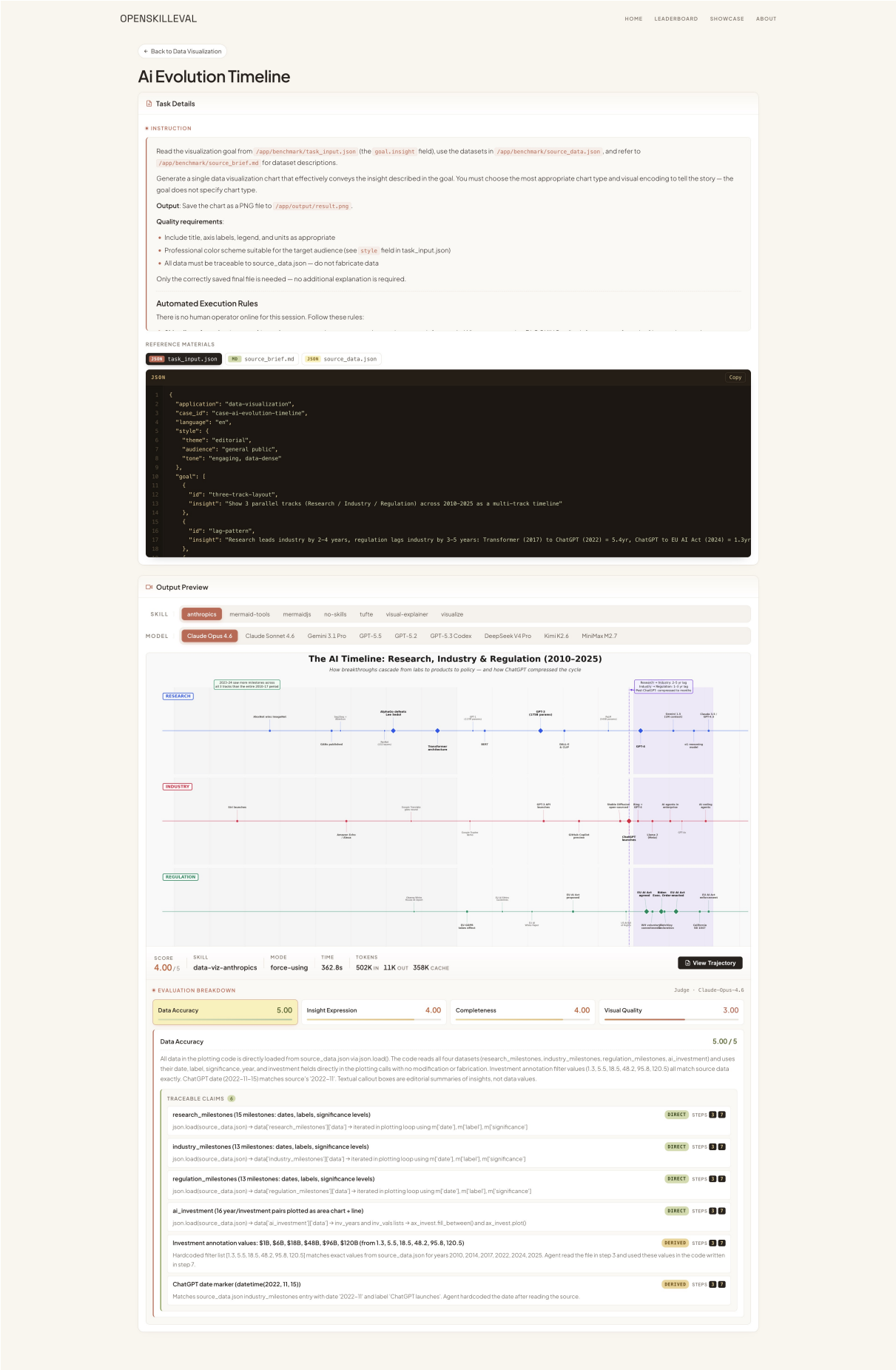


Figure 12: Data Visualization

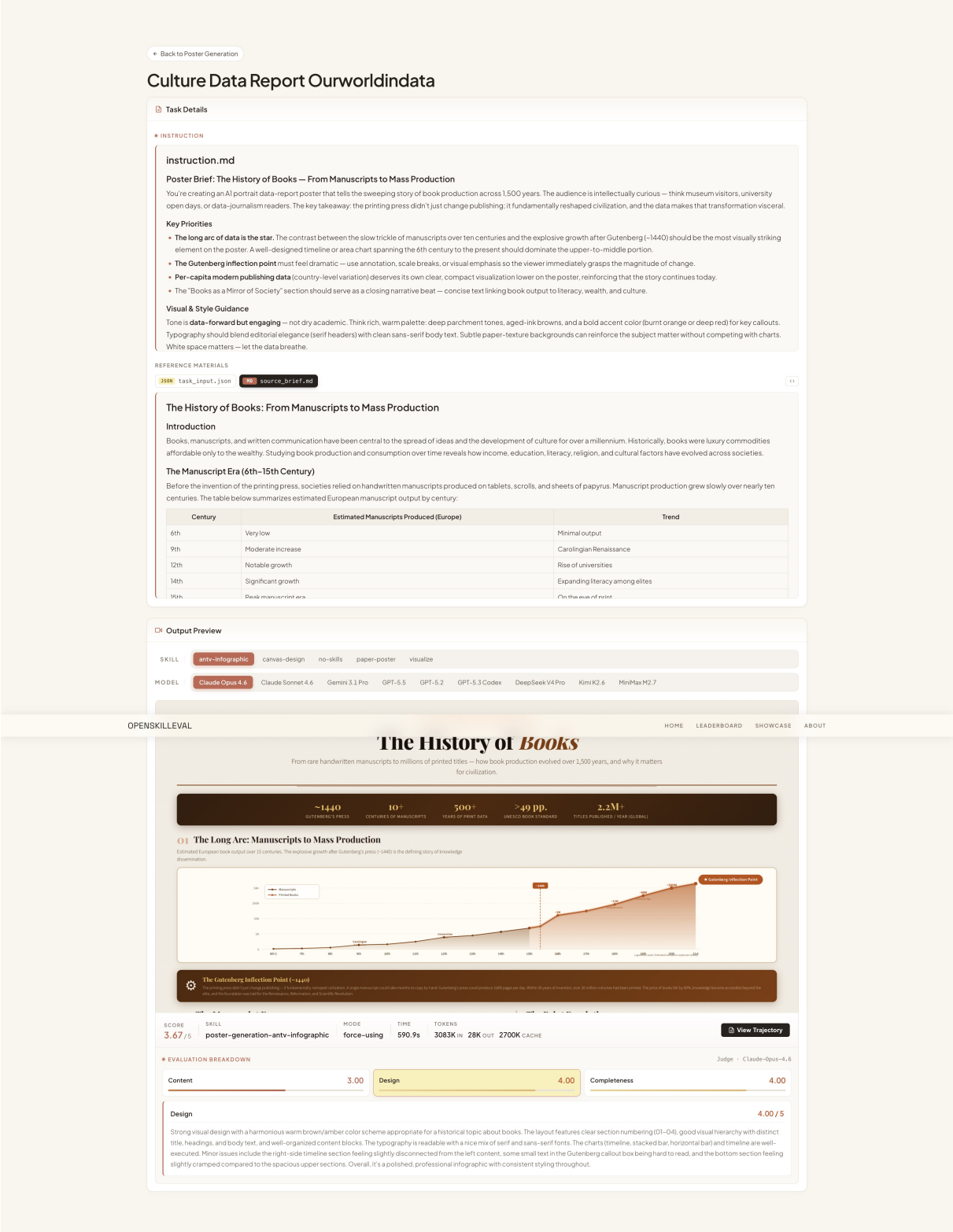


Figure 13: Poster Generation

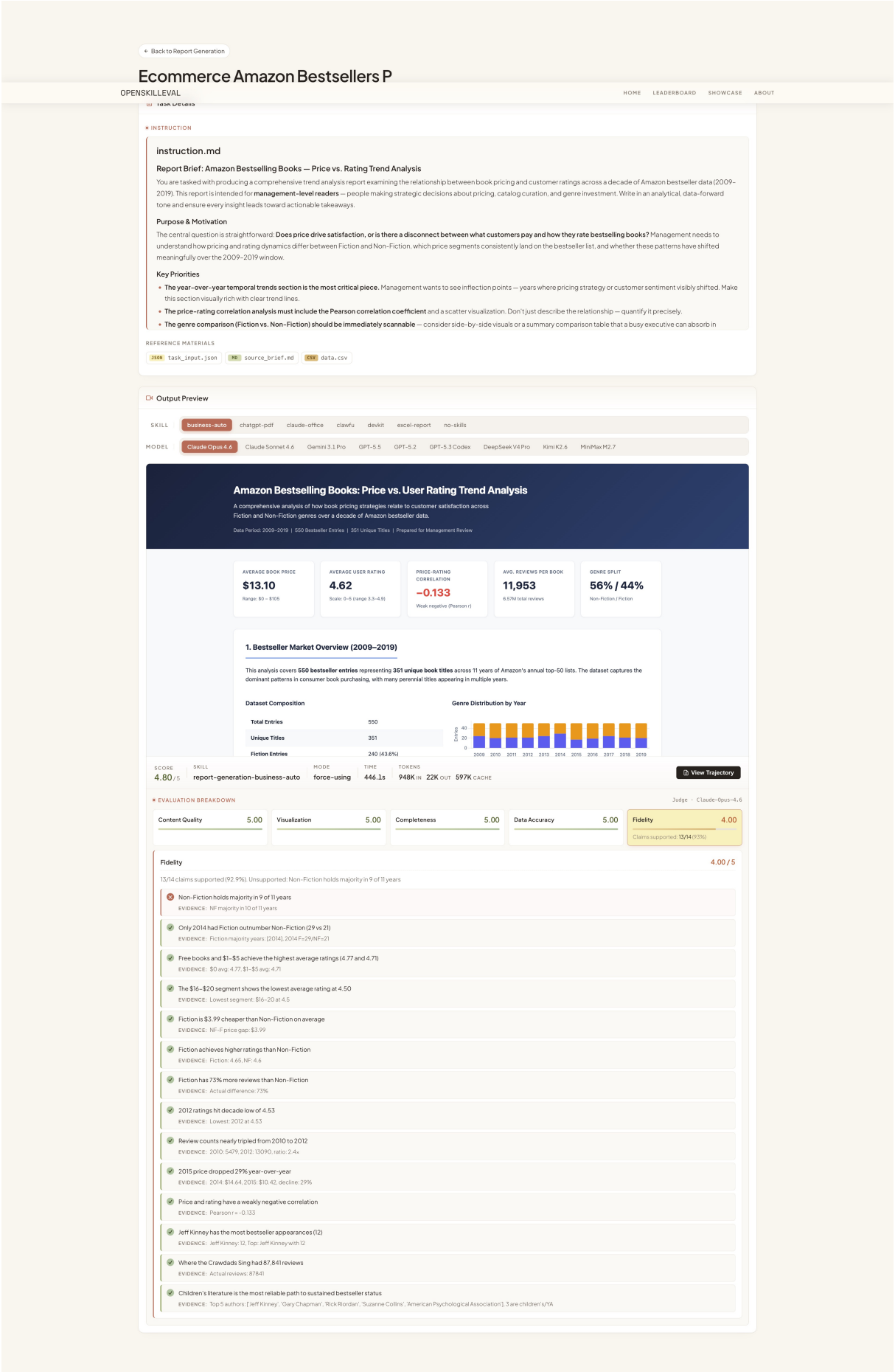


Figure 14: Report Generation

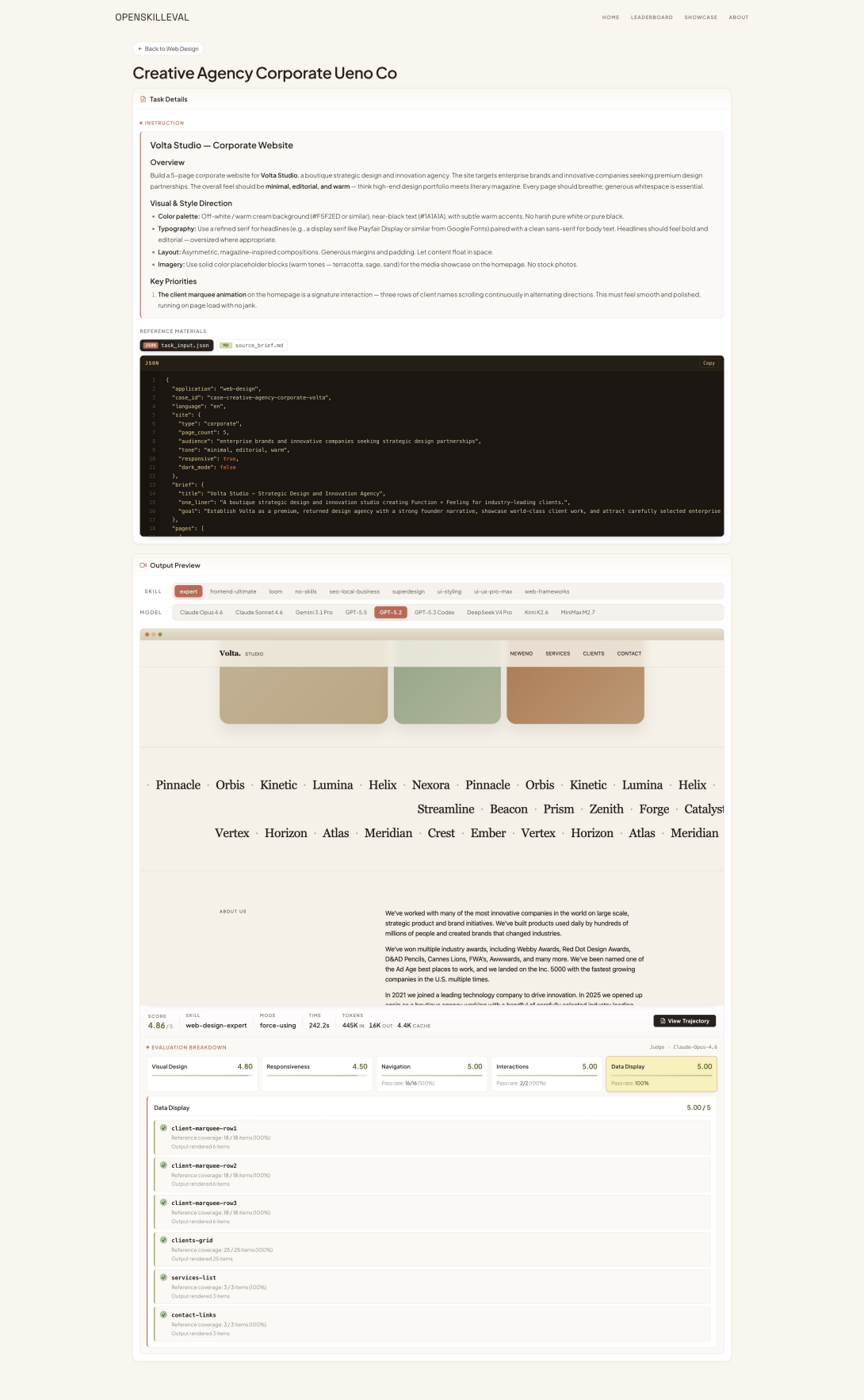


Figure 15: Front-end Web Design

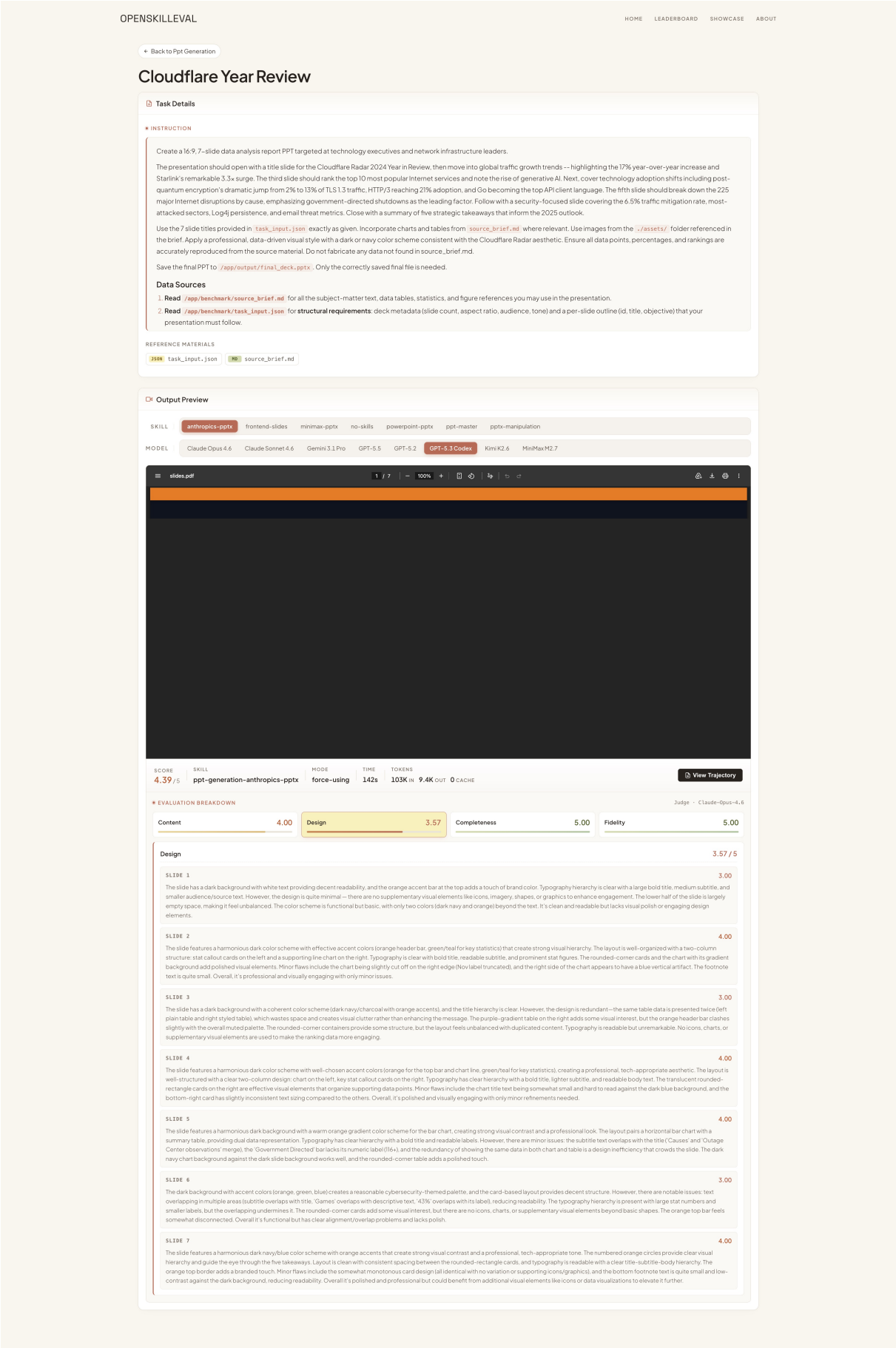


Figure 16: Presentation Generation