PPTArena: A Benchmark for PowerPoint Editing

Michael Ofengenden[1,2⋆], Yunze Man[1], Ziqi Pang[1], Liang-Yan Gui[1], and Yu-Xiong Wang[1]

[1] University of Illinois Urbana-Champaign
[2] University of California, Berkeley

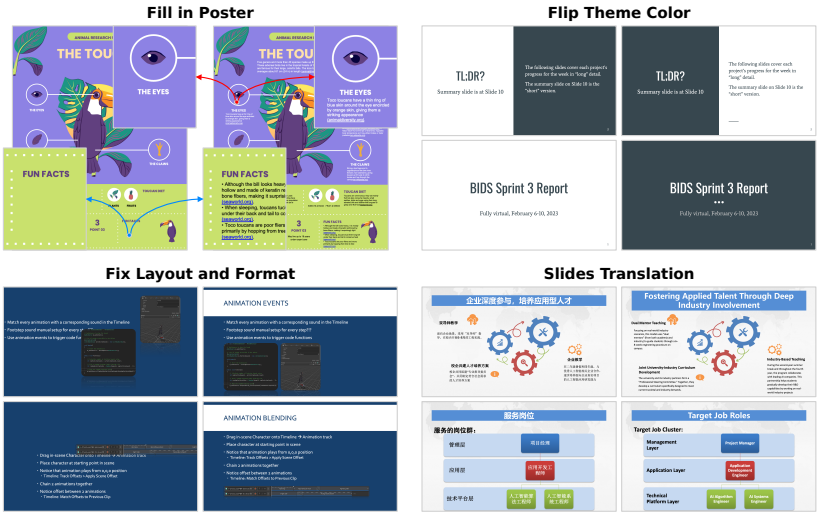


Fig. 1: Examples showing structure-aware, multimodal, and cross-slide reasoning tasks.

Abstract. We introduce PPTArena, a benchmark for PowerPoint editing that evaluates how agents modify real slides from natural-language instructions. Unlike benchmarks that rely on image-PDF renderings or text-to-slide generation, PPTArena features 100 decks with over 1,300 human-curated edits across 2,125 slides, spanning text, charts, animations, and professional master styles. Each edit pairs a ground-truth deck with a target rubric and is scored by two Vision-Language Model (VLM) judges: one rates instruction following from structural diffs, the other visual quality from slide images. On top of this benchmark, we present PPTPilot, a structure-aware agent that plans semantic edit sequences, routes between programmatic tools and deterministic XML operations, and verifies each result in an iterative plan-edit-check loop. PPTPilot outperforms strong VLM-based agents by more than 10 percentage points on compound, layout-sensitive, and cross-slide edits, with large gains in visual fidelity and deck-wide consistency. Despite this, all agents still struggle on long-horizon, document-scale tasks, underscoring how hard reliable PowerPoint editing remains. We publicly release our code: https://github.com/michaelofengend/PPTArena.

⋆ Work done during a FoDOMMaT internship at UIUC.

1 Introduction

As Vision-Language Model (VLM) agents begin to operate software, the capability that matters most in everyday use—editing existing PowerPoint (PPT) decks with precision—remains largely unverified and under-supported [44]. Image- or PDF-based formulations discard deck semantics (formats, shape trees, master layouts), while text-to-slides pipelines emphasize generation and ignore edit-in-place constraints [14, 27]. This gap matters because most decks are refined through revision, not from scratch, making reliable layout reasoning and non-destructive modification the realistic bar for agentic PPT capability. Yet we still lack a benchmark that asks the practical question: *can today's multimodal agents reliably edit existing decks with high instruction fidelity and visual quality?*

Reliable PPT editing is intrinsically hard. Rasterized "image editing" views discard the object- and style-level structure that makes editing precise: fonts and paragraphs, shape geometries, z-order, theme colors, slide masters, and cross-references are all lost once a deck is treated as a bitmap. The same instruction (*e.g.*, "make the subtitle 18pt and align the two logos to the grid") can require multiple coordinated actions across several slides, conditioned on the existing layout and theme. Evaluation is equally subtle: a change can be syntactically valid yet semantically wrong or aesthetically poor. These failure modes are systematically invisible to benchmarks that only check final text strings, API-level diffs, or pixel similarity, and they motivate a benchmark that treats PPT editing as a structured program over deck semantics, with explicit scoring of both instruction following and visual quality.

Our motivation is grounded in how presentations are actually made and maintained. In professional and academic settings, most decks do not begin from a completely blank canvas; they evolve through continuous revision: merging slides from prior talks, adapting templates for new audiences, and polishing visual hierarchy. Editing reveals whether an agent truly understands the structure already present: the agent must locate the correct element, reason about its relationships (alignment, grouping, z-order), and modify it without collateral changes elsewhere. To capture this reality, we introduce **PPTArena**, a benchmark designed for PowerPoint editing on real decks. PPTArena assembles 100 real-world source decks and 2,125 slides into 1,300+ discrete, human-specified edits that range from local text updates to compound, cross-slide transformations, such as deck-wide theme flips, accessibility passes, and multi-step layout repairs.

Unlike generation datasets that can easily output thousands of static templates, agentic editing requires causal, element-level ground-truths for in-place modifications. Creating these high-fidelity edit pairs is an order of magnitude more complex; consequently, our 1,300+ tasks provide substantial statistical power, exceeding the scale of recent multimodal agent benchmarks like OSWorld (369 tasks) [59] or SWE-bench Multimodal (517 visual issues) [61]. Furthermore, to ensure rigorous task depth and structural variety, our 100 source decks were meticulously filtered from over 18,000 candidates across academic, corporate, creative, and multilingual domains. Each edit bundles an initial deck, a fully specified target deck, and a *style-aware rubric* that disambiguates correctness at

the level of content, typography, layout, and color roles. Representative tasks are illustrated in Figure 1, including filling in missing poster content while preserving hierarchy, flipping theme color roles consistently across a deck, fixing layout and format across related slides, and translating slide content while maintaining charts and structure. To our knowledge, PPTArena is the *first* benchmark that (i) treats PPT editing as a structured, causal program over deck semantics, (ii) ships element-level ground-truths, style targets, and error rubrics to disambiguate correctness, and (iii) uses dual instruction-following and visual judges that are extensively validated against human expert ratings to allow diverse configurations of edits while maintaining task coherence.

Complementing the benchmark, we present **PPTPilot**, a structure-aware pilot agent for robust, fine-grained PPT editing. PPTPilot decomposes each natural-language instruction into a sequence of semantic operations, chooses between high-level APIs (*e.g.*, `python-pptx`) and direct XML patching, and validates the outcome against task-specific targets. Two design choices are key: *structure-aware planning*, where the agent parses slide masters, placeholders, shape trees, text, and visual data before editing, and *deterministic execution*, where XML-level patches and strict schemas give exact control over fonts, theme color slots, positions, and master-level changes while programmatic tools handle repetitive global operations (*e.g.*, translation, bulk normalization). An iterative plan-edit-verify loop, coupled with XML validation and visual checks, improves robustness on visually demanding and long-horizon edits.

We evaluate a broad spectrum of baselines on PPTArena, including strong proprietary PPT agents (*e.g.*, ChatGPT Agent [44] and MiniMax Agent [42]), extended-thinking VLM configurations, and ablations of PPTPilot. Even with generous prompting and tool access, existing systems struggle to balance instruction fidelity with visual quality on compound, multi-step edits, and they frequently fail on cross-slide dependencies and master-level changes surfaced by our benchmark. In contrast, PPTPilot achieves substantially higher scores, improving over strong proprietary agents and frontier VLM systems by more than 10 percentage points on compound, layout-sensitive, and cross-slide edits, while maintaining competitive performance on simpler tasks. Nonetheless, PPTPilot and all evaluated agents still fail on hard, visually dependent tasks, suggesting both the difficulty of PPTArena and the headroom for future research.

Our key contributions are threefold: (1) **PPTArena**, a benchmark for agentic PowerPoint editing that (i) operates on deck-native structure rather than rasterized slides, (ii) offers a taxonomy of single- and multi-edit tasks that stress structural grounding, cross-slide consistency, accessibility, and narrative intent, and (iii) pairs each edit with element-level ground-truths, style targets, and a dual-judge protocol (rigorously aligned with human perception) that separately measures instruction fidelity and visual/layout quality, extending beyond prior PPT evaluation setups [14, 19]. (2) **PPTPilot**, a structure-aware pilot agent that plans edits over semantic elements and executes them via a hybrid of high-level programmatic tools and deterministic XML patching, with routing, strict schemas, and iterative verification designed for controllability, reliability, and transparency.

(3) **A comprehensive empirical study** of proprietary agents, open VLMs, and PPTPilot variants on PPTArena, revealing that the benchmark is challenging even for state-of-the-art systems. Ultimately, we hope our work bridges the gap between static visual perception and actionable agentic AI, demonstrating that true multimodal understanding requires agents not just to interpret visual layouts, but to causally manipulate their underlying structures.

2 Related Work

PowerPoint editing sits at the intersection of autonomous agent platforms, productivity automation, and evaluation tooling. We review related work highly relevant to PPTArena across multimodal agentic benchmarks, presentation editing systems, industrial agent frameworks, and LLM-as-judge evaluation [5, 15, 17, 21].
Multimodal agentic and slide benchmarks. General-purpose multimodal benchmarks ensure agents possess the perceptual and reasoning depth required for high-quality edits. A line of benchmarks demands grounding and knowledge integration beyond literal reading [2, 49, 55]. Other datasets stress robustness and preference alignment [25, 64]. More challenging benchmarks aggregate expert-level tasks across multiple disciplines [65, 66]. PowerPoint-centered benchmarks expose how fragile instruction fidelity remains for today's VLMs. PPTC and PPTC-R evaluate multi-turn editing sessions [19, 68], and SlideAudit offers a structured look at design quality, while ANA probes temporal comprehension ignored by static evaluations [26, 69]. Concurrent and independent work [13] also benchmarks agents on PowerPoint, but evaluates GUI-based creation and editing across only 12 source files; in contrast, PPTArena targets in-place editing at an order-of-magnitude larger scale (100 decks, 2,125 slides, 1,300+ edits) with deterministic structural diffs and per-sample style targets.

Broader agentic benchmarks have progressed from controlled settings toward realistic, multimodal environments—spanning the web, native GUIs and code-driven OS control, and industrial tasks with UI randomization [4, 6, 8, 28, 30, 33, 53, 54, 59, 60, 72]—while aggregate leaderboards highlight how evaluator design, scaling laws, and benchmark rigor shape reported capability [15, 17, 21, 23, 36, 38, 41, 74].
Presentation editing. Agentic presentation pipelines blend planning, content synthesis, and low-level manipulation. A line of work explores combinations of generation and refinement, yet each reports brittleness in object targeting, template bias, or cascading errors [14, 27, 46, 62, 70]. Other work highlights the cost of over-reliance on generic templates [27]. Comparative studies confirm API-driven execution outperforms GUI-based approaches for fine-grained control [10, 67], motivating PPTArena's XML-level enforcement and pixel-grounded targets.
Industrial agent and tool-calling. Progress in agent infrastructure illustrates how self-supervised tools expand computer-use competence [20, 22, 48, 58] and perception-driven agents [24, 39, 73]. Open-source orchestration stacks make it easier to compose planners, memory modules, and tool executors, while industrial reports detail production deployments [1, 9, 16, 32, 43, 44, 57]. Concurrent analyses of scaling curves and structured agent engineering [17, 21] warn that larger models

alone do not guarantee reliability, reinforcing PPTArena's focus on judge audits that make agent improvements interpretable.

LLM- and VLM-as-judge evaluation. Reliable evaluation remains a bottleneck as agent tasks grow more open-ended. A line of work studies the feasibility of delegating assessment to specialized VLMs [7, 37, 71]. Follow-up work exposes risks of judge bias, prompting recommendations such as no-free-label baselines, adversarial judge detection, multi-judge audits, and alignment across heterogeneous tasks [18, 31, 35, 64]. Our work adopts these lessons through a dual-judge protocol, separating instruction compliance from visual and layout grading, ensuring that progress measured on PPTArena reflects genuine improvements rather than exploitation of single-model biases.

3 PPTArena Benchmark

PPTArena contains 100 decks with real-world editing instructions spanning 2,125 slides. Each edit bundles an initial deck, a target deck with human-generated ground-truth references, structured textual instructions, and a rubric capturing layout, typography, color, and content requirements. We group the edits into sixteen topical buckets (detailed in Table 1), ensuring that the benchmark stresses both semantic reasoning and low-level formatting fidelity.

3.1 Benchmark Composition and Difficulty

Data sourcing and coverage. We web-scrape over 18,000 PowerPoints (including slides from SlidesCarnival [52], Zenodo [12], SlideShare [50])—the largest open-sourced dataset of PPTs—and convert them into structured JSON traces that capture layout, styling, and content metadata.

Automated filtering retains decks with diverse multimodal assets, which we further combine with a curated internal corpus contributed by literature analysts, biology researchers, and art and design students. From more than 500 hand-reviewed candidates, including 25 decks created from scratch by us, we select the 100 PowerPoints that best span professional, academic, multi-lingual, and art/design genres, ensuring every topic bucket in Table 1 contains challenging exemplars.

Taxonomy-driven task design. Drawing from established principles of presentation design [3, 11, 47, 56], we define five parent categories that decompose into 16 concrete edit types, guaranteeing coverage from low-level typography to master-edit workflows. Following [40], we favor fewer but richer edits, so the taxonomy is used to balance high-difficulty scenarios rather than to inflate totals. Tasks range from simple text replacements to multi-edit, multimodal reasoning problems. For instance, Figure 1 illustrates a cross-modal editing task that requires matching images to captions across slides using both visual and textual cues; many editing tasks straddle multiple taxonomy buckets, so category counts exceed 100.

Table 1: Taxonomy of 16 editing operations in PPTArena. The five major categories: Content, Layout, Styling, Interactivity, and Structure, encompass operations from basic text manipulation to advanced master-level edits and accessibility compliance.

Category	Edit Types
Content	**(1)** Text & Typography • **(2)** Shapes & Drawing • **(3)** Images & Pictures • **(4)** Tables • **(5)** Charts • **(6)** SmartArt & Diagrams • **(7)** Audio & Video
Layout	**(8)** Alignment, Distribution, Grid, Grouping, Z-order • **(9)** Slide Layouts & Placeholders
Styling	**(10)** Themes (colors, fonts, effects), Background • **(11)** Master-level edits (Slide/Notes Masters)
Interactivity	**(12)** Animations (entrance, emphasis, exit, paths, timing) • **(13)** Slide Transitions • **(14)** Hyperlinks
Structure	**(15)** Slide/Section/Order Mgmt., Slide Numbers, Headers/Footers, Notes • **(16)** Comments/Review, Accessibility (alt text, reading order, contrast)

PPTArena distinguishes itself from prior benchmarks through its emphasis on *edit difficulty* across four key dimensions: multi-step reasoning depth, cross-slide dependencies, semantic understanding requirements, and long-horizon planning complexity. While existing resources such as PPTC-R [68] and T2US [27] focus on short, template-bound edits, PPTArena intentionally concentrates difficulty into fewer but richer scenarios to expose real-world failures.

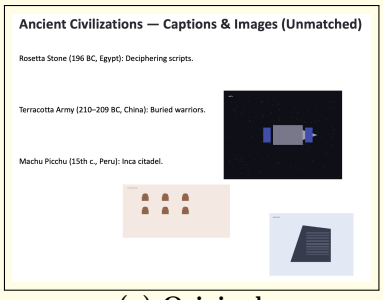
Difficulty distribution. Table 2 quantifies this concentration of difficulty. We flag a case as *cross-slide* when it requires coordinated edits spanning at least two slides, and as *high-diff* when it involves cross-slide dependencies or strong visual-textual reasoning such as chart remapping or translation. Structure and Interactivity edits require the longest programs (17.8–18.9 ops) over the most slides (10–11), driving the highest cross-slide (56–70%) and high-diff (71–75%) rates, while even Content, Layout, and Styling keep roughly a third of cases cross-slide and 41–52% high-diff. PPTArena thus targets multi-slide, multimodal reasoning rather than inflating counts with single-slide tweaks.

3.2 Comparison with Prior Benchmarks

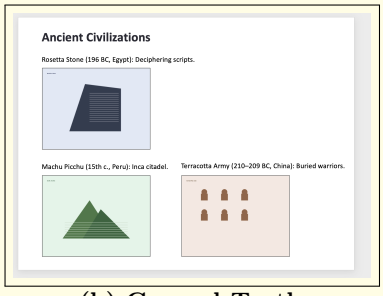
Table 3 quantitatively compares PPTArena against prior benchmarks. PPTArena demonstrates substantially higher complexity with an average of 13.4 operations per edit and 8.3 slides per edit, with 32% of edits involving cross-slide dependencies and 28% requiring visual-textual reasoning. Prior benchmarks suffer from key limitations: PPTC-R relies on synthetically generated decks that lack real-world visual richness and complexity, while T2US artificially inflates its size by rewording prompts for identical tasks rather than introducing genuine task diversity [27, 34, 68].

Cross-Slide Image-Caption Correlation

"Category": *Layout*
"Prompt": *1. For every slide, match each image to its correct caption. 2. After pairing them, rearrange the pairs to create a balanced layout. 3. Update each slide's title correspondingly. 4. Finally, make sure all images are the same size, 3.2 inches wide by 2.4 inches high.*



(a) Original



(b) Ground-Truth

Configure Speaker Notes

"Category": *Content, Structure*
"Prompt": *Slide 2 contains speaker notes for the other slides. Please move the text from the text boxes on slide 2 to the speaker notes of the appropriate slides, then delete slide 2.*

(c) Original slide 2

(d) Ground-Truth slide 6

Fig. 2: Representative edits from PPTArena. **Top:** Cross-slide image-caption matching requires semantic understanding to correlate visual and textual content. **Bottom:** Configuring speaker notes requires aligning narration, speaker notes, and visual panels across slides, forcing the agent to reason about cross-slide correspondences before deleting the staging slide.

Table 2: Challenge distribution in PPTArena. Per-category averages of atomic edit operations and slides touched, with the share of *cross-slide* and *high-diff* cases.

Category	Cases	Avg. ops	Avg. slides	Cross-slide	High-diff.
Content	67	11.3	6.8	26%	41%
Layout	29	15.0	8.9	34%	52%
Styling	29	13.4	9.4	30%	48%
Structure	15	17.8	11.2	56%	71%
Interactivity	4	18.9	10.5	70%	75%
All tags	144	13.4	8.3	32%	49%

Table 3: Comparison with prior benchmarks. PPTArena emphasizes richer, multi-operation, cross-slide, and multimodal tasks.

Metric	PPTC-R	T2US	PPTArena
Task Complexity			
Avg. operations per edit	2.9	1.2	**13.4**
Avg. slides per edit	1.3	1.2	**8.3**
Cross-slide dependencies	21%	5%	**32%**
Text-visual dependencies	×	1.3%	**28%**
Benchmark Design			
Human-created decks	×	✓	✓
Python-created decks	✓	×	✓
Ground-truth provided	✓	×	✓
Cross-platform compatibility	✓	×	✓
Advanced Requirements			
Accessibility constraints	×	×	✓
External knowledge	×	×	✓
Complex multimodal tasks	×	×	✓

In contrast, PPTArena combines both human-created and Python-generated decks, provides ground-truth for deterministic evaluation, and maintains cross-platform compatibility. It uniquely incorporates accessibility constraints, external knowledge requirements, and complex multimodal tasks that emphasize multi-step reasoning, cross-slide dependencies, and long-horizon planning. This design exposes failure modes that remain hidden on simpler tasks and provides a rigorous testbed for evaluating the next generation of agentic systems. We demonstrate two representative samples in Figure 2.

3.3 VLM-as-Judge Evaluation Protocol

Instruction following (IF) and visual quality (VQ). Our evaluation framework is designed to move beyond simple pixel- or code-level diffs, which fail to capture the semantic and aesthetic goals of PPT editing. We instead measure an agent's performance on two fundamental axes: *Instruction Following* and *Visual Quality* [51], both scored by expert VLM judges on an integer scale from 0 (Failure) to 5 (Perfect).

(1) *Instruction Following (IF)* measures the agent's semantic and logical adherence to the user's prompt. It assesses *what* was done, such as correctly identifying and moving content, applying the right formatting, or fulfilling all sub-tasks in a complex command.

(2) *Visual Quality (VQ)* measures the *aesthetic and professional polish* of the resulting slide. It assesses *how* the changes were implemented, focusing on layout, alignment, typography, color harmony, and overall visual appeal, independent of the instruction's logical fulfillment.

By combining the two metrics together, our PPTArena covers the common user requirements from either content aspects (IF) or aesthetic aspects (VQ).

Per-sample rubric: style target. A core challenge in PPT editing is the immense variation across decks, layouts, and design habits. There is no universal

rubric that can reliably score every instruction, which separates our setting from common LLM judges used in question-answering. Our solution is to generate a fine-grained, per-sample style target that specifies all crucial structural and visual requirements for that editing task. For example, if the user prompt is "Overhaul this rock-cycle presentation with ..., reorganizing the rock types into three columns on slide 2, and replace slide 3's wall of text with ...," then the corresponding style-target rubric would spell out the exact ground-truth with hyper-specific constraints, such as: *"... must have a geology photo occupying ... Slide 2 columns must be labeled ... Slide 3 centers a fully labeled cycle diagram linking the three rock types with ..."*.

To provide trustworthy style targets, we combine automatic generation together with exhaustive human verification. Specifically, we send the JSON summaries and screenshots of the PPT's ground-truth and original decks to GPT-5 [43] to generate style targets. Both ground-truth and the original slide are provided as input so that the generation of the style target can precisely understand the desired outcome. Then each style target is manually verified for correctness and faithfulness to the editing instructions and PPT context.

Dual-judge framework for reliable evaluation. To ensure reliable scoring, we employ a dual-judge framework that separately conducts the evaluation for instruction following and visual quality (as in Figure 3), each implemented as a separate VLM, *e.g.*, GPT-5.2 [43]. To further enhance the judge's capability for IF and VQ, respectively, we selectively provide them contexts that align best with the evaluation target, in addition to the style target mentioned above.

(1) *Instruction Following Judge*: This judge receives *only* the structured data diffs (*e.g.*, JSON and XML summaries) between the original, predicted, and ground-truth slides. Such inputs force the judge to concentrate at the content level and carefully inspect whether the desired content changes are correct.

(2) *Visual Quality Judge*: This judge is responsible for visual aesthetics, so it receives *only* the rendered screenshots of the predicted and ground-truth slides. Its context is engineered to focus purely on aesthetics, comparing the visual execution of alignment, layout, and style against the rubric.

To further improve the reliability, especially for multi-slide edits, we use Structural Similarity Index Measure (SSIM) screening to forward only the slides with salient changes to the visual judge, so that it concentrates better on the edits without being overwhelmed by the enormous contexts.

Comparison with judges in related benchmarks. Our evaluation methodology marks a significant advance over existing benchmarks. Prior work, such as PPTC-R [19], primarily relies on API-level "diffs". While useful, this approach is brittle and cannot detect critical semantic errors, such as an agent copying the correct text but to the *wrong slide*. Our IF Judge, by operating on structured summaries, is explicitly designed to capture such logical failures. Furthermore, while other benchmarks like T2US [27] also use VLM-as-judge, their reliance on prompting without a strong rubric leads to noisy and unreliable ratings. PPTArena designs the style target to mitigate such issues by providing a rigorous and reproducible foundation for scoring.

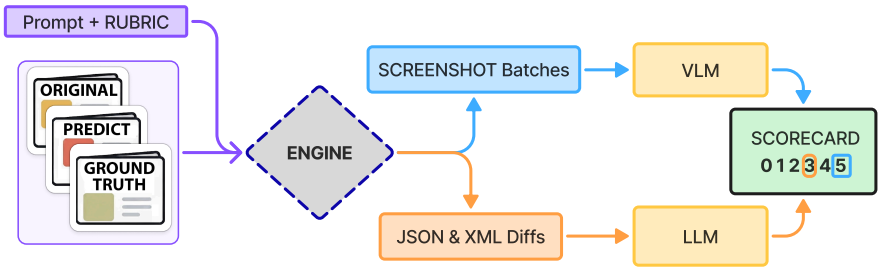


Fig. 3: Our VLM-as-judge paradigm. To maximize the reliability of existing VLMs, we employ two separate judges. The visual quality (VQ) judge primarily comprehends the PPT screenshots for visual understanding, while the instruction-following (IF) judge focuses on structured data to analyze the contents.

To conclude, our dual-judge, rubric-grounded architecture is essential for measuring the complex, multi-step reasoning required for PPT editing and for analyzing the subtle failure modes that simpler benchmarks would miss.

4 An Effective PPT Editing Agent: PPTPilot

We introduce PPTPilot, an agent for presentation editing, whose simple architecture yields surprisingly effective results, even outperforming proprietary products like OpenAI's ChatGPT Agent [44] (details in Sec. 5.1). Our design is built on two key insights. **(1)** The primary challenge in this domain is not solely intent recognition, but reliability and precision. PowerPoint files are built on the brittle Office Open XML (OOXML) format, which is highly intolerant to malformed or "hallucinated" VLM outputs, and therefore requires specialized formats and contexts suitable for PPT editing. **(2)** We found that no single editing modality, *e.g.*, purely relying on the XML format, is sufficient. A robust agent must be capable of intelligently selecting the optimal tool and editing interface for a given task.

PPTPilot's core design is a *dual-path* architecture, emphasizing the ability to handle editing queries via either programmatic tools or direct XML editing (Figure 4). This hybrid design enables our agent to address a wide range of editing queries reliably and in a principled way.

Programmatic editing. Utilizing `python-pptx` to edit PPTs programmatically—a method commonly adopted in prior work [14, 27, 45, 62, 70]—scripts the edits by generating code (bottom of Figure 4). This approach is highly effective for repetitive, well-defined, and content-centric operations, such as performing a "find-and-replace" across all slides or translating text. However, it lacks the fine-grained control required for complex structural modifications (*e.g.*, altering slide masters, themes, or specific layout geometries).

Direct XML editing. To address the limitations of the programmatic path in structural and visual editing scenarios, we have equipped PPTPilot with a second

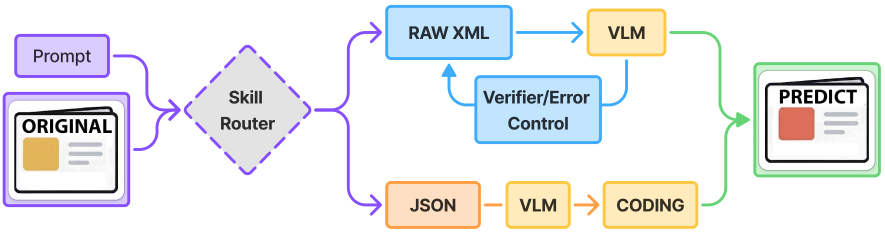


Fig. 4: Our PPTPilot paradigm. Our key insight highlights a combination of two different editing skills: functional code and direct XML edits to control the fine-grained structural elements. A "Skill Router" determines which skill is more suitable for a query. And then the corresponding VLM executes the edits via either route. Notably, the editing operations can also be enhanced with reflection, trading time for more reliable editing. For the router we use a fast LLM (GPT-5 nano or Gemini-3.0 flash), and then GPT-5.2 for our VLM edit calls.

skill: the ability to directly read, parse, and rewrite raw OOXML files (*e.g.*, `slide1.xml`, `theme.xml`), as shown in the top half of Figure 4. This approach provides the precision required for structured contexts, as the VLMs can directly manipulate fine-grained properties like the specific positions of elements. Since OOXML encodes most of the information in a PPT, the XML path provides a unified interface well-aligned with existing VLMs for PPT editing. However, the long context and strict format requirements of XML make it challenging to perform precise edits, especially when modifications span a large number of slides, in which case the programmatic approach is significantly more reliable.

Skill routing. To determine which editing skills to adopt for a specific user query, we employ an LLM that routes the query to the proper editing skills, as the beginning of branching in Figure 4. Upon receiving a user instruction, this decider analyzes the prompt combined with the presentation's structure, including the screenshots and contents. Based on this analysis, it routes the task to either the programmatic path or the direct XML editing path. Across PPTArena, the router dispatches 66% of edits to the programmatic path and 34% to the direct XML path (Supp. Sec. G.1).

Self-correction with reflection. Finally, we acknowledge the complexity of PPT editing, which indicates the challenge of correct edits in a single try. Inspired by representative agents such as ReACT [63], we introduce an iterative reflection path into the PPTPilot, so that it can gradually refine its predictions. The agent proposes an edit to the XML files, which is rendered temporarily to a PPT file. Then a verifier model assesses the output PPT according to the original instructions and provides feedback for failures. In this way, the agent produces an updated PPT edit based on the feedback and is able to correct for its own errors.

Table 4: PPTArena matched 25-case subset evaluation. All systems here are evaluated on the same 25-deck, 206-edit subset.

		PPTPilot		Gemini CLI		ChatGPT		ChatGPT Agent		MiniMax Agent		Kimi-K2.6		PPTAgent	
Category	**Decks**	IF↑	VQ↑	IF↑	VQ↑	IF↑	VQ↑	IF↑	VQ↑	IF↑	VQ↑	IF↑	VQ↑	IF↑	VQ↑
Content	19	**2.00**	**1.95**	1.84	1.53	1.05	1.35	1.80	1.50	1.10	0.75	1.90	1.15	0.00	0.00
Layout	7	1.86	**2.29**	1.57	1.57	**2.00**	**2.29**	1.14	0.71	0.71	0.71	1.86	0.57	0.00	0.00
Styling	5	**2.40**	1.60	2.00	2.40	1.83	**2.67**	0.83	1.67	1.00	1.00	1.33	1.17	0.00	0.00
Structure	3	1.47	2.00	0.67	2.00	0.33	1.33	**2.00**	**2.33**	0.67	1.33	1.00	0.33	0.00	0.00
Interactivity	1	**3.00**	**2.00**	0.00	0.00	1.00	1.00	0.00	0.00	2.00	1.00	0.00	0.00	0.00	0.00
Total	25	**1.87**	**1.91**	1.78	1.71	1.12	1.56	1.68	1.60	1.04	0.84	1.68	1.00	0.00	0.00

Despite the simplicity of our design, we find it effective and efficient for PPT editing when compared against existing agent products and frameworks. We hope our PPTPilot can serve as a baseline for research into PPT editing.

5 Experiments

VLM-as-Judge. We evaluate with the strongest vision-language models, Gemini 3.1 Pro [16] and GPT-5.2 [43]. The main comparison reports the GPT-5.2 judge (Tables 4 and 5); a cross-judge robustness check under Gemini 3.1 Pro is summarized in Sec. 5.2, with the full per-system breakdown and implementation details in the supplementary (Supp. Table A).
Agent Baselines. We evaluate a wide range of PPT agents against PPTPilot: extended-thinking ChatGPT [43], Gemini-CLI [16], ChatGPT-Agent [44] (which drives a simulated desktop and terminal), and MiniMax Agent, marketed as a multimodal "PPT helper." We additionally include the open-weight Kimi-K2.6 [29] as a non-proprietary reference point. PPTAgent [70] and Paper2Poster [45] both fail on all tasks.
Subset Evaluation. Because several proprietary products enforce strict rate limits and high costs (*e.g.*, ChatGPT and MiniMax Agent [42]), we also evaluate on a budget-limited subset of 25 decks: the 20 hardest cases plus 5 added for breadth. Subset details, per-system budgets, and the ≈$700 total cost are in the supplementary (Supp. Secs. D and I).

5.1 Performance on PPTArena

Table 5 summarizes system-level performance across the full benchmark. PPTPilot attains the strongest overall results with an instruction-following score of 2.57 and a visual-quality score of 2.69.
Baseline system performance. ChatGPT performs well on straightforward content edits and light styling, yet drops on tasks requiring visual-text alignment, cross-slide reasoning, or deck-wide structural constraints. ChatGPT Agent shows somewhat stronger visual performance but struggles with multi-step logical instructions. Its relative edge on *Structure* edits stems from GUI-level control: PowerPoint natively handles slide masters, footers, and section dividers, whereas PPTPilot can over-edit master XML when a change is slide-local. MiniMax

Table 5: PPTArena full-benchmark evaluation

			PPTPilot		Gemini CLI		ChatGPT	
Category	**Decks**	**Edits**	IF↑	VQ↑	IF↑	VQ↑	IF↑	VQ↑
Content	67	887	**2.54**	**2.75**	1.30	2.54	2.03	2.20
Layout	29	504	**2.64**	**2.38**	1.07	1.78	2.08	2.19
Styling	29	413	2.32	**2.72**	0.91	1.89	**2.41**	2.44
Structure	15	182	**2.43**	**2.95**	1.32	2.27	1.73	1.93
Interactivity	4	30	3.00	**3.00**	0.00	0.00	**3.25**	2.75
Total	100	1340	**2.57**	**2.69**	1.21	1.98	2.07	2.22

Agent, despite being marketed as a presentation tool, underperforms consistently: it is notable only in the visual-layout category, trailing PPTPilot and ChatGPT Agent elsewhere. PPTAgent highlights the fragility of one-shot, generation-driven pipelines: its outputs diverge substantially from the original structure, breaking preservation requirements and failing all tasks under our rubric.

Runtime efficiency, latency, and key design principles. We report end-to-end wall-clock latency per task (Table 6); runs that fail to produce a valid PPTX within our cap are timeouts. PPTPilot completes edits quickly in a single pass, while ChatGPT Agent often enters long verification loops and can hit the cap; a 3-iteration reflection loop stays competitive in latency while improving reliability. Together, these results indicate that reliable PPT editing hinges on three properties embodied in PPTPilot: (i) structure-aware planning over deck semantics, (ii) a hybrid execution model routing between programmatic APIs and deterministic OOXML operations, and (iii) an iterative refine-and-verify loop that stabilizes long-horizon edits. We break down these techniques in Table 7 (full study in Supp. Sec. E.3).

Table 6: End-to-end latency per task under each method's original environment.

Method	**Latency (min) ↓**
ChatGPT Agent	4–30
MiniMax Agent	3
PPTPilot (1-pass)	1.5
PPTPilot (3-loop)	3

Ablating PPTPilot's design. Table 7 isolates the three properties above. Forcing a single execution path collapses performance: XML-only reaches only 0.95 IF (it struggles with deck-wide operations), and `python-pptx`-only reaches 2.06 IF (it lacks fine-grained structural control). Hybrid routing recovers most of the gap (2.36 IF), and the iterative refine-and-verify loop lifts the full system to 2.84 IF / 3.21 VQ, with most corrections landing by the second pass. The top block validates the dual-judge protocol: a single judge over all signals is over-lenient (VQ 4.26, a 33% IF/VQ gap), and dropping structured diffs inflates scores further (3.76/4.54), whereas the dual judge with diffs yields the calibrated, internally consistent scores we report throughout.

Table 7: PPTPilot ablation on PPTArena. Average instruction fidelity (IF) and visual quality (VQ) across judge configurations (top block) and executor variants (bottom block). Hybrid routing plus the reflection loop is essential; the dual judge with structured diffs is the only stable evaluation regime.

Configuration	IF↑	VQ↑
Judge configuration		
Single VLM judge (all signals)	2.31	4.26
Dual judge (no diffs)	3.76	4.54
Dual judge with diffs	2.36	2.40
PPTPilot executor variants		
XML-only	0.95	2.85
`python-pptx`-only	2.06	2.73
Hybrid (no refinement)	2.36	2.69
Hybrid + Loop (3×)	**2.84**	**3.21**

5.2 VLM-as-Judge Reliability and Human Alignment

Code-level metrics fail to capture nuanced visual semantics such as occlusion, alignment, and layout balance, so we evaluate with two specialized VLM judges—one for *Instruction Following* (IF) and the other for *Visual Quality* (VQ). To validate that these track human perception, we run a blind study on a stratified 10% subset across all editing categories, with 25 human experts rating IF and VQ on the judges' ordinal scale. Table 8 shows strong human-judge rank and linear correlations, confirming the judges' reliability.

We also assess pipeline stability by running each judge five times per example on identical inputs, reporting the fraction of examples with majority agreement and the standard deviation of the discrete scores (as percentages). Table 8 shows that VQ judging is exceptionally consistent, and both metrics give stable, reproducible signals. A page-wise control study confirms batching does not distort VQ: an independent Gemini 3.1 Pro judge scoring 105 matched pairs one page at a time agrees within one point on 8 of 10 cases (mean VQ 1.88 page-wise vs. 2.00 batched; Pearson r=0.917).

Table 8: Human alignment and judge consistency (IF/VQ) on a stratified 10% subset with 25 expert raters.

Metric	IF	VQ
Pearson (r) ↑	0.72	0.81
Spearman (ρ) ↑	0.65	0.80
Kendall (τ) ↑	0.52	0.71
Consistency (5 runs per example)		
Majority agreement ↑	78%	95%
Std. dev. ↓	15.4%	9%

Cross-judge robustness. To confirm the leaderboard is not an artifact of a single judge, we re-score every system with an independent Gemini 3.1 Pro judge (full per-category table in Supp. Table A). Absolute values shift slightly, but the ranking is stable: PPTPilot retains its lead in both IF (2.45) and VQ (2.74) on the full benchmark, ahead of ChatGPT (1.97/2.03) and Gemini CLI (1.92/2.15), and keeps its lead on the hard subset.

5.3 Human Evaluation of PPTPilot vs. Proprietary Agents

Evaluators consistently rate PPTPilot above the proprietary ChatGPT Agent and MiniMax Agent baselines; the newly rated MiniMax Agent (IF/VQ = 1.75/1.85) remains far below PPTPilot (3.86/3.81, Table 9). These human-grounded results corroborate our automated findings (Sec. 5.1).

Table 9: Blind human scores (25 expert raters) on the stratified subset.

Method	IF ↑	VQ ↑
MiniMax Agent	1.75	1.85
ChatGPT Agent	3.17	2.82
PPTPilot	**3.86**	**3.81**

What the human study adds. Under direct preference the gaps widen: PPTPilot leads ChatGPT Agent by 0.69 IF / 0.99 VQ and MiniMax by over two points on both axes (Table 9), with raters flagging the deck-wide-constraint and text–image alignment failures our rubric targets (Supp. Sec. H).

6 Conclusion

We introduce PPTArena and PPTPilot for evaluating and improving PowerPoint editing in agentic multimodal systems. PPTArena grounds evaluation in real decks with structured rubrics and dual judges for instruction fidelity and visual quality. PPTPilot's dual-path plan-edit-verify design improves reliability over strong proprietary agents and frontier VLMs. Yet agents still fail on complex, long-horizon, multi-modal tasks indicating that robust PPT editing remains far from solved. We hope PPTArena and PPTPilot seed future work on reliable editing of the documents people actually use.

Limitations and future work. Open directions include conversational refinement for under-specified intent, cross-application workflows (live charts, document-to-deck synthesis), and hyper-specialized domains beyond our current taxonomy; agents also remain brittle on edits coupling visual, spatial, and cross-slide reasoning (Supp. Secs. H and K).

Acknowledgements. This work was supported in part by NSF under Grants 2106825 and 2519216; the DARPA Young Faculty Award; the ONR Grant N00014-26-1-2099; the NIFA Award 2020-67021-32799; the NSF REU Site Program under Award 2447843 (FoDOMMaT: The Future of Discovery: Training Students to Build and Apply Open-Source Machine Learning Models and Tools); the Apple AIML Academic Research Program; the Amazon-Illinois Center on AI for Interactive Conversational Experiences; the Capital One Illinois Center for Generative AI Safety, Knowledge Systems, and Cybersecurity; and the Toyota Research Institute. This work used computational resources, including the NCSA Delta and DeltaAI supercomputers through allocations CIS230012, CIS230013, CIS240133, and CIS240387 from the Advanced Cyberinfrastructure Coordination Ecosystem: Services & Support (ACCESS) program; the TACC Frontera supercomputer, Amazon Web Services (AWS), and OpenAI API through the National Artificial Intelligence Research Resource (NAIRR) Pilot; and Google Cloud research credits through the Gemini Academic Program.

References

1. Adobe: Adobe introduces new AI assistant in Adobe Express that unlocks creativity for all with conversational creation and editing. Adobe Newsroom press release (Oct 2025), https://news.adobe.com/news/2025/10/adobe-max-2025-express-ai-assistant, Adobe MAX 2025, October 28, 2025. Accessed: 2026-06-30
2. Alfarano, A., Venturoli, L., Negueruela del Castillo, D.: VQArt-Bench: A semantically rich VQA benchmark for art and cultural heritage. In: ICCVW (2025). https://doi.org/10.1109/ICCVW69036.2025.00761
3. Alley, M., Schreiber, M., Ramsdell, K., Muffo, J.: How the design of headlines in presentation slides affects audience retention. Technical Communication **53**(2) (2006)
4. Anupam, S., Brown, D., Li, S., Wong, E., Hassani, H., Bastani, O.: BrowserArena: Evaluating LLM agents on real-world web navigation tasks. arXiv preprint arXiv:2510.02418 (2025), https://arxiv.org/abs/2510.02418
5. Bandi, A., Kongari, B., Naguru, R., Pasnoor, S., Vilipala, S.V.: The rise of agentic AI: A review of definitions, frameworks, architectures, applications, evaluation metrics, and challenges. Future Internet **17**(9) (2025). https://doi.org/10.3390/fi17090404
6. Bonatti, R., Zhao, D., Bonacci, F., Dupont, D., Abdali, S., Li, Y., Lu, Y., Wagle, J., Koishida, K., Bucker, A., Jang, L.K., Hui, Z.: Windows Agent Arena: Evaluating multi-modal OS agents at scale. In: ICML (2025), https://proceedings.mlr.press/v267/bonatti25a.html
7. Chen, D., Chen, R., Zhang, S., Wang, Y., Liu, Y., Zhou, H., Zhang, Q., Wan, Y., Zhou, P., Sun, L.: MLLM-as-a-judge: Assessing multimodal LLM-as-a-judge with vision-language benchmark. In: ICML (2024), https://proceedings.mlr.press/v235/chen24h.html
8. Deng, X., Gu, Y., Zheng, B., Chen, S., Stevens, S., Wang, B., Sun, H., Su, Y.: Mind2Web: Towards a generalist agent for the web. In: NeurIPS Datasets and Benchmarks (2023), https://proceedings.neurips.cc/paper_files/paper/2023/file/5950bf290a1570ea401bf98882128160-Paper-Datasets_and_Benchmarks.pdf
9. Derouiche, H., Brahmi, Z., Mazeni, H.: Agentic AI frameworks: Architectures, protocols, and design challenges. arXiv preprint arXiv:2508.10146 (2025), https://arxiv.org/abs/2508.10146
10. Ding, Z., Wang, X., Chen, J., Kristensson, P.O., Shen, J.: Prompt-driven agentic video editing system: Autonomous comprehension of long-form, story-driven media. arXiv preprint arXiv:2509.16811 (2025), https://arxiv.org/abs/2509.16811
11. Duarte, N.: slide:ology: The Art and Science of Creating Great Presentations. O'Reilly Media, Sebastopol, CA (2008)
12. European Organization for Nuclear Research (CERN), OpenAIRE: Zenodo. Research data repository (2013), https://zenodo.org/, general-purpose open research repository launched May 2013 by OpenAIRE and CERN. Accessed: 2026-06-30
13. Gandhi, A., Suryanarayanan, V., Shaik, F., Anwar, R.H., Desai, S., Nguyen, T.Q., Raza, M.T., Chowdhary, V., Neubig, G.: PPTArena: A benchmark for computer-use agents on PowerPoint tasks. OpenReview preprint (2025), https://openreview.net/forum?id=Dl1S4EvFwh
14. Ge, J., Wang, Z.Z., Zhou, X., Peng, Y.H., Subramanian, S., Tan, Q., Sap, M., Suhr, A., Fried, D., Neubig, G., Darrell, T.: AutoPresent: Designing structured visuals from scratch. In: CVPR (2025). https://doi.org/10.1109/CVPR52734.2025.00276

15. Ge, Y., Hua, W., Mei, K., Ji, J., Tan, J., Xu, S., Li, Z., Zhang, Y.: OpenAGI: When LLM meets domain experts. In: NeurIPS Datasets and Benchmarks (2023), https://proceedings.neurips.cc/paper_files/paper/2023/file/1190733f217404edc8a7f4e15a57f301-Paper-Datasets_and_Benchmarks.pdf
16. Gemini Team, Google DeepMind: Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. arXiv preprint arXiv:2507.06261 (2025), https://arxiv.org/abs/2507.06261
17. Gonzalez-Pumariega, G., Tu, V., Lee, C.L., Yang, J., Li, A., Wang, X.E.: The unreasonable effectiveness of scaling agents for computer use. arXiv preprint arXiv:2510.02250 (2025), https://arxiv.org/abs/2510.02250
18. Gu, J., Jiang, X., Shi, Z., Tan, H., Zhai, X., Xu, C., Li, W., Shen, Y., Ma, S., Liu, H., Wang, S., Zhang, K., Wang, Y., Gao, W., Ni, L., Guo, J.: A survey on LLM-as-a-judge. arXiv preprint arXiv:2411.15594 (2024), https://arxiv.org/abs/2411.15594
19. Guo, Y., Zhang, Z., Liang, Y., Zhao, D., Duan, N.: PPTC benchmark: Evaluating large language models for PowerPoint task completion. In: Findings of ACL (2024). https://doi.org/10.18653/v1/2024.findings-acl.514
20. Gur, I., Furuta, H., Huang, A.V., Safdari, M., Matsuo, Y., Eck, D., Faust, A.: A real-world WebAgent with planning, long context understanding, and program synthesis. In: ICLR (2024), https://arxiv.org/abs/2307.12856
21. Hassan, A.E., Li, H., Lin, D., Adams, B., Chen, T.H., Kashiwa, Y., Qiu, D.: Agentic software engineering: Foundational pillars and a research roadmap. arXiv preprint arXiv:2509.06216 (2025), https://arxiv.org/abs/2509.06216
22. He, H., Yao, W., Ma, K., Yu, W., Dai, Y., Zhang, H., Lan, Z., Yu, D.: WebVoyager: Building an end-to-end web agent with large multimodal models. In: ACL (2024). https://doi.org/10.18653/v1/2024.acl-long.371
23. Hendrycks, D., Song, D., Szegedy, C., Lee, H., Gal, Y., Brynjolfsson, E., Li, S., Zou, A., Levine, L., Han, B., Fu, J., Liu, Z., Shin, J., Lee, K., Mazeika, M., Phan, L., Ingebretsen, G., Khoja, A., Xie, C., Salaudeen, O., Hein, M., Zhao, K., Pan, A., Duvenaud, D., Li, B., Omohundro, S., Alfour, G., Tegmark, M., McGrew, K., Marcus, G., Tallinn, J., Schmidt, E., Bengio, Y.: A definition of AGI. arXiv preprint arXiv:2510.18212 (2025), https://arxiv.org/abs/2510.18212
24. Hsiao, Y.C., Zubach, F., Baechler, G., Sunkara, S., Carbune, V., Lin, J., Wang, M., Zhu, Y., Chen, J.: ScreenQA: Large-scale question-answer pairs over mobile app screenshots. In: NAACL (2025). https://doi.org/10.18653/v1/2025.naacl-long.477
25. Huang, C., Maneechotesuwan, B., Chopra, S., Kira, Z.: FRAMES-VQA: Benchmarking fine-tuning robustness across multi-modal shifts in visual question answering. In: CVPR (2025). https://doi.org/10.1109/CVPR52734.2025.00370
26. Jiang, Y., Xue, Y., Kang, Y., Zheng, P., Peng, J., Wu, F., Xu, C.: Animation needs attention: A holistic approach to slides animation comprehension with visual-language models. arXiv preprint arXiv:2507.03916 (2025), https://arxiv.org/abs/2507.03916
27. Jung, K., Cho, H., Yun, J., Yang, S., Jang, J., Choo, J.: Talk to your slides: High-efficiency slide editing via language-driven structured data manipulation. In: ACL (2026), https://arxiv.org/abs/2505.11604, accepted
28. Kapoor, R., Butala, Y.P., Russak, M., Koh, J.Y., Kamble, K., AlShikh, W., Salakhutdinov, R.: OmniACT: A dataset and benchmark for enabling multimodal generalist autonomous agents for desktop and web. In: ECCV (2024). https://doi.org/10.1007/978-3-031-73113-6_10

29. Kimi Team: Kimi K2: Open agentic intelligence. arXiv preprint arXiv:2507.20534 (2025), https://arxiv.org/abs/2507.20534
30. Koh, J.Y., Lo, R., Jang, L., Duvvur, V., Lim, M., Huang, P.Y., Neubig, G., Zhou, S., Salakhutdinov, R., Fried, D.: VisualWebArena: Evaluating multimodal agents on realistic visual web tasks. In: ACL (2024). https://doi.org/10.18653/v1/2024.acl-long.50
31. Krumdick, M., Lovering, C., Reddy, V., Ebner, S., Tanner, C.: No free labels: Limitations of LLM-as-a-judge without human grounding. arXiv preprint arXiv:2503.05061 (2025), https://arxiv.org/abs/2503.05061
32. LangChain, Inc.: LangGraph. Software framework (2024), https://github.com/langchain-ai/langgraph, accessed: 2026-06-30
33. Lee, J., Min, T., An, M., Hahm, D., Lee, H., Kim, C., Lee, K.: Benchmarking mobile device control agents across diverse configurations. In: CoLLAs (2025), https://proceedings.mlr.press/v330/lee26a.html
34. Lee, S., Kim, S., Park, S.H., Kim, G., Seo, M.: Prometheus-Vision: Vision-language model as a judge for fine-grained evaluation. In: Findings of ACL (2024). https://doi.org/10.18653/v1/2024.findings-acl.672
35. Li, D., Tan, Z., Zhao, C., Jiang, B., Huang, B., Ma, P., Alnaibari, A., Shu, K., Liu, H.: Who's your judge? on the detectability of LLM-generated judgments. arXiv preprint arXiv:2509.25154 (2025), https://arxiv.org/abs/2509.25154
36. Li, W., Lin, J., Jiang, Z., Cao, J., Liu, X., Zhang, J., Huang, Z., Chen, Q., Sun, W., Wang, Q., Lu, H., Qin, T., Zhu, C., Yao, Y., Fan, S., Li, X., Wang, T., Liu, P., Zhu, K., Zhu, H., Shi, D., Wang, P., Guan, Y., Tang, X., Liu, M., Jiang, Y.E., Yang, J., Liu, J., Zhang, G., Zhou, W.: Chain-of-Agents: End-to-end agent foundation models via multi-agent distillation and agentic RL. arXiv preprint arXiv:2508.13167 (2025), https://arxiv.org/abs/2508.13167
37. Liu, M., Zhang, W.: Is your video language model a reliable judge? In: ICLR (2025), https://arxiv.org/abs/2503.05977
38. Liu, X., Yu, H., Zhang, H., Xu, Y., Lei, X., Lai, H., Gu, Y., Ding, H., Men, K., Yang, K., Zhang, S., Deng, X., Zeng, A., Du, Z., Zhang, C., Shen, S., Zhang, T., Su, Y., Sun, H., Huang, M., Dong, Y., Tang, J.: AgentBench: Evaluating LLMs as agents. In: ICLR (2024), https://openreview.net/forum?id=zAdUB0aCTQ
39. Ma, Z., Zhang, B., Zhang, J., Yu, J., Zhang, X., Zhang, X., Luo, S., Wang, X., Tang, J.: SpreadsheetBench: Towards challenging real world spreadsheet manipulation. In: NeurIPS Datasets and Benchmarks (2024), https://openreview.net/forum?id=KYxzmRLF6i
40. Maia Polo, F., Weber, L., Choshen, L., Sun, Y., Xu, G., Yurochkin, M.: tinyBenchmarks: Evaluating LLMs with fewer examples. In: ICML (2024), https://proceedings.mlr.press/v235/maia-polo24a.html
41. Mialon, G., Fourrier, C., Wolf, T., LeCun, Y., Scialom, T.: GAIA: a benchmark for general AI assistants. In: ICLR (2024), https://openreview.net/forum?id=fibxvahvs3
42. MiniMax: MiniMax M2 & Agent: Ingenious in simplicity. MiniMax News (blog) (2025), https://www.minimax.io/news/minimax-m2, accessed: 2026-06-30
43. OpenAI: GPT-5 system card. System card (2025), https://cdn.openai.com/gpt-5-system-card.pdf, accessed: 2025-10-19
44. OpenAI: Introducing ChatGPT agent: Bridging research and action. OpenAI blog post (2025), https://openai.com/index/introducing-chatgpt-agent/, accessed: 2026-06-22

45. Pang, W., Lin, K.Q., Jian, X., He, X., Torr, P.: Paper2Poster: Towards multimodal poster automation from scientific papers. In: NeurIPS Datasets and Benchmarks (2025), https://arxiv.org/abs/2505.21497
46. Qian, K., Li, W., Sun, T., Wang, W., Luo, W.: DocRefine: An intelligent framework for scientific document understanding and content optimization based on multimodal large model agents. arXiv preprint arXiv:2508.07021 (2025), https://arxiv.org/abs/2508.07021
47. Reynolds, G.: Presentation Zen: Simple Ideas on Presentation Design and Delivery. New Riders, Berkeley, CA, 2nd edn. (2011)
48. Schick, T., Dwivedi-Yu, J., Dessì, R., Raileanu, R., Lomeli, M., Hambro, E., Zettlemoyer, L., Cancedda, N., Scialom, T.: Toolformer: Language models can teach themselves to use tools. In: NeurIPS (2023), https://arxiv.org/abs/2302.04761
49. Schwenk, D., Khandelwal, A., Clark, C., Marino, K., Mottaghi, R.: A-OKVQA: A benchmark for visual question answering using world knowledge. In: ECCV (2022). https://doi.org/10.1007/978-3-031-20074-8_9
50. Scribd, Inc.: SlideShare: Online presentation sharing community. Website (2024), https://www.slideshare.net/, accessed: 2026-06-30
51. Sim, M.Y., Zhang, W.E., Dai, X., Fang, B.: Can VLMs actually see and read? a survey on modality collapse in vision-language models. In: Findings of ACL (2025). https://doi.org/10.18653/v1/2025.findings-acl.1256
52. SlidesCarnival: SlidesCarnival: Free PowerPoint and Google Slides templates. Website (2024), https://www.slidescarnival.com/, accessed: 2026-06-30
53. Takahashi, J., Moteki, A., Uchida, A., Masui, S., Yang, F., Uchino, K., Song, Y., Bisk, Y., Neubig, G., Kusajima, I., Watanabe, Y., Ishida, H., Nakagawa, K., Jiang, S.: FieldWorkArena: Agentic AI benchmark for real field work tasks. In: ICPR (2026), https://arxiv.org/abs/2505.19662, accepted
54. Tian, S., Zhang, Z., Chen, L., Liu, Z.: MMInA: Benchmarking multihop multimodal internet agents. In: Findings of ACL (2025). https://doi.org/10.18653/v1/2025.findings-acl.703
55. Tran, D.T., Tran, T.K., Hauswirth, M., Le Phuoc, D.: ReasonVQA: A multi-hop reasoning benchmark with structural knowledge for visual question answering. In: ICCV (2025), https://arxiv.org/abs/2507.16403
56. Williams, R.: The Non-Designer's Design Book: Design and Typographic Principles for the Visual Novice. Peachpit Press, San Francisco, CA, 4th edn. (2015)
57. Wu, Q., Bansal, G., Zhang, J., Wu, Y., Li, B., Zhu, E., Jiang, L., Zhang, X., Zhang, S., Liu, J., Awadallah, A.H., White, R.W., Burger, D., Wang, C.: AutoGen: Enabling next-gen LLM applications via multi-agent conversation. In: COLM (2024), https://openreview.net/forum?id=tEAF9LBdgu
58. Wu, Z., Han, C., Ding, Z., Weng, Z., Liu, Z., Yao, S., Yu, T., Kong, L.: OS-Copilot: Towards generalist computer agents with self-improvement. arXiv preprint arXiv:2402.07456 (2024), https://arxiv.org/abs/2402.07456
59. Xie, T., Zhang, D., Chen, J., Li, X., Zhao, S., Cao, R., Toh, J.H., Cheng, Z., Shin, D., Lei, F., Liu, Y., Xu, Y., Zhou, S., Savarese, S., Xiong, C., Zhong, V., Yu, T.: OSWorld: Benchmarking multimodal agents for open-ended tasks in real computer environments. In: NeurIPS Datasets and Benchmarks (2024), https://papers.nips.cc/paper_files/paper/2024/hash/5d413e48f84dc61244b6be550f1cd8f5-Abstract-Datasets_and_Benchmarks_Track.html
60. Xu, T., Chen, L., Wu, D.J., Chen, Y., Zhang, Z., Yao, X., Xie, Z., Chen, Y., Liu, S., Qian, B., Yang, A., Jin, Z., Deng, J., Torr, P., Ghanem, B., Li, G.: CRAB: Cross-environment agent benchmark for multimodal language model agents. In: Findings of ACL (2025). https://doi.org/10.18653/v1/2025.findings-acl.1113

61. Yang, J., Jimenez, C.E., Zhang, A.L., Lieret, K., Yang, J., Wu, X., Press, O., Muennighoff, N., Synnaeve, G., Narasimhan, K.R., Yang, D., Wang, S.I., Press, O.: SWE-bench multimodal: Do AI systems generalize to visual software domains? In: ICLR (2025), https://openreview.net/forum?id=riTiq3i21b
62. Yang, Y., Jiang, W., Wang, Y., Song, Y., Wang, Y., Zhang, C.: Auto-Slides: An interactive multi-agent system for creating and customizing research presentations. In: ICME (2026), https://arxiv.org/abs/2509.11062, accepted
63. Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., Cao, Y.: ReAct: Synergizing reasoning and acting in language models. In: ICLR (2023), https://arxiv.org/abs/2210.03629
64. Yasunaga, M., Zettlemoyer, L., Ghazvininejad, M.: Multimodal RewardBench: Holistic evaluation of reward models for vision language models. arXiv preprint arXiv:2502.14191 (2025), https://arxiv.org/abs/2502.14191
65. Yue, X., Ni, Y., Zhang, K., Zheng, T., Liu, R., Zhang, G., Stevens, S., Jiang, D., Ren, W., Sun, Y., Wei, C., Yu, B., Yuan, R., Sun, R., Yin, M., Zheng, B., Yang, Z., Liu, Y., Huang, W., Sun, H., Su, Y., Chen, W.: MMMU: A massive multi-discipline multimodal understanding and reasoning benchmark for expert AGI. In: CVPR (2024). https://doi.org/10.1109/CVPR52733.2024.00913
66. Yue, X., Zheng, T., Ni, Y., Wang, Y., Zhang, K., Tong, S., Sun, Y., Yu, B., Zhang, G., Sun, H., Su, Y., Chen, W., Neubig, G.: MMMU-Pro: A more robust multi-discipline multimodal understanding benchmark. In: ACL (2025). https://doi.org/10.18653/v1/2025.acl-long.736
67. Zhang, C., He, S., Li, L., Qin, S., Kang, Y., Lin, Q., Rajmohan, S., Zhang, D.: API agents vs. GUI agents: Divergence and convergence. arXiv preprint arXiv:2503.11069 (2025), https://arxiv.org/abs/2503.11069
68. Zhang, Z., Guo, Y., Liang, Y., Zhao, D., Duan, N.: PPTC-R benchmark: Towards evaluating the robustness of large language models for PowerPoint task completion. In: Findings of EMNLP (2024). https://doi.org/10.18653/v1/2024.findings-emnlp.722
69. Zhang, Z.J., Chen, R., Zhong, M., Wobbrock, J.O.: SlideAudit: A dataset and taxonomy for automated evaluation of presentation slides. In: UIST (2025). https://doi.org/10.1145/3746059.3747736
70. Zheng, H., Guan, X., Kong, H., Zhang, W., Zheng, J., Zhou, W., Lin, H., Lu, Y., Han, X., Sun, L.: PPTAgent: Generating and evaluating presentations beyond text-to-slides. In: EMNLP (2025). https://doi.org/10.18653/v1/2025.emnlp-main.728
71. Zheng, L., Chiang, W.L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., Lin, Z., Li, Z., Li, D., Xing, E.P., Zhang, H., Gonzalez, J.E., Stoica, I.: Judging LLM-as-a-judge with MT-bench and chatbot arena. In: NeurIPS Datasets and Benchmarks (2023), https://arxiv.org/abs/2306.05685
72. Zhou, S., Xu, F.F., Zhu, H., Zhou, X., Lo, R., Sridhar, A., Cheng, X., Ou, T., Bisk, Y., Fried, D., Alon, U., Neubig, G.: WebArena: A realistic web environment for building autonomous agents. In: ICLR (2024), https://openreview.net/forum?id=oKn9c6ytLx
73. Zhu, R., Cheng, X., Liu, K., Zhu, B., Jin, D., Parihar, N., Xu, Z., Gao, O.: SheetMind: An end-to-end LLM-powered multi-agent framework for spreadsheet automation. arXiv preprint arXiv:2506.12339 (2025), https://arxiv.org/abs/2506.12339
74. Zhu, Y., Jin, T., Pruksachatkun, Y., Zhang, A., Liu, S., Cui, S., Kapoor, S., Longpre, S., Meng, K., Weiss, R., Barez, F., Gupta, R., Dhamala, J., Merizian, J., Giulianelli, M., Coppock, H., Ududec, C., Kellermann, A., Sekhon, J.S., Steinhardt, J., Schwettmann, S., Narayanan, A., Zaharia, M., Stoica, I., Liang, P., Kang, D.:

Establishing best practices in building rigorous agentic benchmarks. In: NeurIPS Datasets and Benchmarks (2025), https://arxiv.org/abs/2507.02825