VLM-SlideEval: Evaluating VLMs on Structured Comprehension and Perturbation Sensitivity in PPT

Hyeonsu B. Kang **Emily Bao** **Anjan Goswami**
PowerPoint AI, Microsoft Inc., San Francisco, CA 94103
{hyeonsukang, baoemily, anjangoswami}@microsoft.com

Abstract

Vision-language models (VLMs) are increasingly used to evaluate multimodal content, including presentation slides, yet their slide-specific understanding remains underexplored despite their growing role as critics in agentic, model-forward pipelines. We introduce **VLM-SlideEval**, an evaluation framework that probes VLMs along three axes: (1) element-level extraction from slide images aligned to ground truth; (2) robustness to controlled perturbations in geometry, style, and text; and (3) higher-level comprehension, such as recovering a deck's narrative order from shuffled slides. Using publicly available decks from Zenodo[1], we standardize ground-truth element metadata from PowerPoint XML and live renderings into a unified, verifiable schema. Empirically, VLMs underperform on pixel-accurate extraction and show non-trivial agreement, fidelity, and consistency under controlled perturbations, while performing better on single-slide content understanding; however, they do not reliably capture narrative structure across slides. These results highlight the limits of current VLMs for slide evaluation and motivate calibrated, critic-in-the-loop evaluators that drive iterative refinement and selection in agentic pipelines.

1 Introduction

Presentation slides are a primary vehicle for conveying structured ideas across domains ranging from education to scientific communication to corporate decision-making. Automatic evaluation of slide quality and content understanding is an emerging and pronounced need, particularly in light of advances in *agentic, model-forward* slide generation [1, 2]. While prior work on document analysis has focused on optical character recognition (OCR) [3, 4, 5] and XML-based parsing [6], these approaches are brittle when slides are only available as rendered images, and are limited to low-level layout information without reasoning about higher-level semantics. In contrast, vision-language models (VLMs) promise a unified mechanism for parsing slide content directly from images while also supporting tasks that require semantic or narrative comprehension.

Despite the promise, it remains unclear to what extent current VLMs truly comprehend presentation slides. On one hand, VLMs may struggle with precise pixel-level tasks such as identifying bounding boxes, font attributes, or alignment, since they may not have been directly trained on raw presentation rendering pipelines or large scale OCR data of slide presentations. On the other hand, VLMs may excel at higher-level understanding, such as identifying the role of slide elements (*e.g.,* title, subtitle, body text), inferring content hierarchy, or reasoning over narrative flow in a deck. Understanding these trade-offs is crucial for designing reliable and scalable evaluation pipelines that utilize VLMs.

We introduce **VLM-SlideEval** as a first-class *critic* in agentic, model-forward pipelines and systematically probe VLM slide comprehension. Our contributions are threefold. First, we curate a diverse dataset of PowerPoint decks and extract ground-truth geometry, style, and text via a pipeline combining PowerPoint XML with rasterized renders. Second, we design protocols for low-level fidelity and structured comprehension, including element-wise Hungarian alignment and refinement-relevant probes of judge reliability (variance, sensitivity) and robustness via controlled perturbations

[1] https://zenodo.org; HF viewer: https://huggingface.co/datasets/Forceless/Zenodo10K/viewer/default/pptx

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Evaluating the Evolving LLM Lifecycle - Benchmarks, Emergent Abilities, and Scaling.

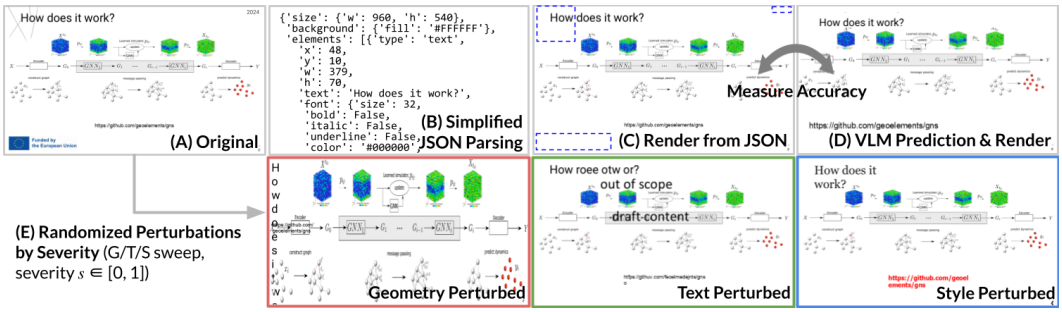


Figure 1: **Evaluation Task Examples:** Top: From an original slide (A), we parse a simplified schema JSON (Table 1) (B), reconstruct a normalized slide (C; blue dashed boxes show theme-embedded content omitted by the schema). A VLM predicts the schema from the re-rendered slide (D), and we score accuracy. Bottom: We subsample 100 decks, retain slides with ≥ 3 visible elements (234 slides total), and apply perturbations to geometry, text, and style with severity $s \in [0, 1]$ (larger s means stronger changes; details in §3). Perturbed slides are then used for VLM quality evaluation and sensitivity analyses (§4).

to geometry, style, and text. Third, we extend evaluation to deck-level narrative by asking VLMs to reorder shuffled slides, assessing coherence.

Applying VLM-SlideEval, we surface clear limits and strengths. VLMs struggle with pixel-accurate extraction and show behavioral divergence under controlled perturbations, yet they competently extract structured content on single slides while remaining unreliable for deck-level narrative. These findings caution against over-reliance on current VLMs for fine-grained slide evaluation and motivate more calibrated critic-in-the-loop refinement and selection gates for agentic pipelines.

2 Related Work

Calibrated VLM evaluators are increasingly critical in agentic, model-forward pipelines: they guide candidate selection, drive iterative refinement at inference time, and even supply reward signals for training. Recent work shows verifier-guided decoding that improves performance without weight updates [7], generalist multimodal judges used both as LMM-as-a-Judge and as reward models [8], actor-critic loops that critique and correct reasoning [9], and refinement-centric benchmarks plus standardization frameworks that emphasize granular measurement [10, 11]. Concurrently, Image2Struct benchmarks VLM image reconstruction on webpages, LaTex, and musical scores [12]. This motivates a slide-native, *verifiable* evaluator that produces actionable signals at pixel, element, and deck levels.

Yet VLM evaluation remains challenging. Open-ended judging often relies on incomplete visual context and fuzzy rubrics, yielding inconsistent scores [13], while models hallucinate and make perceptual errors in visually grounded reasoning [14]. Under *controlled manipulations* and counterfactuals, VLMs may inject priors unsupported by pixels and show limited sensitivity to fine-grained changes [15, 16]. Robustness studies further find text corruptions especially damaging, lightweight adapters sometimes rivaling full fine-tuning, and broader axes (fairness, toxicity, multilinguality) underexplored [17, 18].

Slide presentations sit within multimodal document understanding, where *structured parsing* underpins both comprehension and authoring. Prior work has explored language-driven manipulation of slide *objects* (not pixels) for faster, faithful editing [19], OCR-free pretraining for screenshots and UI/text layouts that improves element-level parsing [20], and automatic extraction of deck structure for role identification and accessibility [21]. In parallel, systems that generate slides from long-form documents highlight the need for *scalable, slide-specific evaluation* [2, 22].

Unlike work that omits a slide-native evaluator, relies on QA proxies, or focuses robustness on charts/UIs, *VLM-SlideEval* provides a slide-specific framework that couples pixel-accurate alignment to PPT-native ground truth with slide-relevant perturbations and deck ordering, *positioning the evaluator as a critic for agentic pipelines*.

3 Method

Data Source. We sample 100 English-dominant ($\geq 70\%$ by `langid` [23]) `.pptx` decks from Zenodo10K (legacy `.ppt` excluded), totaling 1,948 slides, with CC-BY 4.0 license (Summary statistics in Appendix A, Table 2).

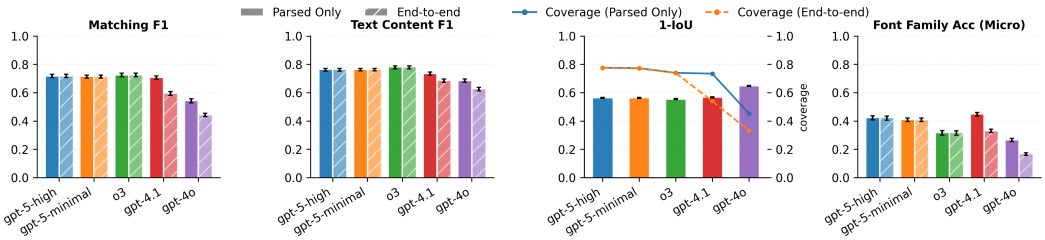


Figure 2: Parsed-only (solid) vs. e2e (hatched) with coverage (*i.e.,* fraction of ground truth instances evaluated for the metric; lines). `o3`/`gpt-5` lead on Matching F1 (0.71-0.72) and Text Content F1 (0.76-0.78); `o3` best in geometry (1-IoU 0.55). Font Family Accuracy is low overall (max 0.42). More results in Fig. 7, Appendix F.

Ground Truth Element *geometry*, *content*, and *style* are extracted from PowerPoint XML and post-layout rendering. We parse static XML and then query the COM (Component Object Model) API after a layout pass to recover effective font metrics and tight text bounds (mitigating AutoFit and container/tight-box discrepancies). Elements are stored in a standardized schema with explicit units (Appendix A, Table 1).

VLM Parsing & GT Matching. Slides are rasterized to PNG and sent with a fixed 960×540px coordinate frame; we test five VLMs (via Azure) to return JSON validated against our schema (invalid JSON counts as a parse failure). Each slide is run ($N = 3$) times (low temperature), and metrics are reported per-run and pooled. Predictions are aligned to GT via Hungarian matching (*cf.* [24, 25, 26, 27, 28]) with a blended cost (1-IoU, center/size difference; text adds content distance) and an acceptance gate; details in Appendix C.

Perturbation Synthesis. *Seeds.* From the same 100 decks we manually select slides well-preserved by the schema and with at least a minimal complexity, ≥ 3 visible text elements, yielding 234 seeds; the reconstructed slide serves as the clean baseline. *Severity knobs.* We generate controlled degradations along *geometry*, *text*, and *style*, parameterized by a single severity $s \in \{0, 0.1, \ldots, 1.0\}$. Magnitudes (*e.g.,* pixel offsets, font-size factors) and event probabilities (*e.g.,* drop/insert text boxes) increase monotonically with s; randomness is seeded per (slide, axis, s). Exports use a Node.js-based PPTX builder and headless rendering. From the 7,722 original+perturbed slides in total (hyperparameters in App. D), we subsample up to 50 slides per severity per axis for evaluation.

Manipulation Check. We assess whether increasing severity $s \in [0, 1]$ yields orderly and proportional degradation using (i) *adjacent POA* (POA_{adj} := the fraction of consecutive severity steps where y^* does not decrease - and (ii) the *mean absolute calibration error* (MACE) to the identity $y^* = s$, on the normalized $[0, 1]$ scale. Empirically, POA is high (5-pt ≈ 0.95; 100-pt ≈ 0.80) with moderate calibration (overall MACE ≈ 0.34).

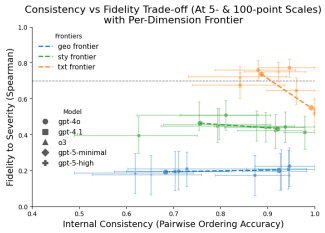
Analysis & Measures We evaluate: (i) *parseability* (schema-valid JSON rate); (ii) end-to-end (e2e) and parsed-only *extraction quality* on matched elements (geometry, content, style); (iii) *narrative ordering* (deck reordering; Kendall's τ, Spearman's ρ); and (iv) *perturbation sensitivity* - R^2, POA and $\text{Spearman}(\text{severity}, y^*)$ - comparing different evaluator scales and models. We report bootstrap 95% CIs where appropriate. Full metric definitions and evaluator prompts appear in Appendix E.

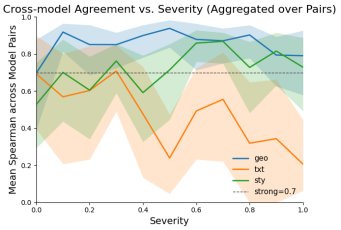
4 Results

We benchmark five VLMs (Azure API) on three main tasks: 1) element-level extraction from slides, 2) behavior under controlled perturbations, and 3) narrative understanding via slide re-ordering.

Slide Parseability. Parse success declines with slide complexity for GPT-4.1 (about 93% for simple slides with ≤ 8 elements, 72.1% for (8-16], 32.8% for (16-32], and 18.2% for ≥ 32 elements). GPT-4o follows a similar trend but with an earlier decline: about 88.0% for ≤ 8, 57.6% for (8-16], 45.8% for (16-32], with a small (noisy) uptick to 66.7% at ≥ 32 ($N = 66$). In contrast, o3 and the GPT-5 variants remain effectively at ceiling across all bins (99.5%+). See Fig. 6.

Element Prediction Accuracy. Across headline metrics (Fig. 2), o3 and the GPT-5 variants lead under e2e. *Matching F1*: Parsed → e2e performance drops ($\Delta \approx 0.12$ for GPT-4.1 and GPT-4o), with o3 achieving the highest e2e F1 score (0.72), followed by GPT-5 (0.71-0.72), vs. GPT-4.1 (0.59) and GPT-4o (0.44). *Text Content F1*: o3 0.78 (best), GPT-5 0.76, GPT-4.1/GPT-4o 0.69/0.63. *Geometry* (1-IoU; lower better): o3 (best, 0.55), GPT-5 (0.56), GPT-4.1 (0.57), GPT-4o (worst, 0.65). E2e coverage is limited, especially for GPT-4o (0.33) and GPT-4.1 (0.54) vs the rest (0.74-0.78) *Styling*





(a) **Consistency-fidelity frontier** per dimension. Consistency is POA_{adj} and fidelity is Spearman (severity, y^*) (higher better). Geometry/style show no fidelity gain but lower consistency when moving from 5- to 100-pt scale; text trades fidelity with consistency.

(b) **Cross-model agreement vs. severity**. Spearman agreement across model pairs by severity buckets. Geometry/style pairs often exceed 0.80-0.90; text is lowest (best text pair $\overline{\rho} \approx 0.55$), indicating limited interchangeability on text.

Figure 3: **Evaluation results of model behavior under controlled perturbations.**

(Font Family Acc.): overall low (0.17-0.42), with GPT-5-high highest (0.42) and GPT-4o lowest (0.17). Detailed metrics and parsed-only comparisons appear in Table 4 and Fig. 7 (App. § F.2).

Behavior Under Controlled Perturbations - Scale correspondence. Within each model, an *isotonic link* maps 5-point scores to 100-point scores with high fidelity: $R^2 \in [0.85, 0.89]$ across models ($p = 0.001$), with GPT-4.1 the tightest ($\text{RMSE} = 0.075$) and others close (*e.g.,* GPT-5-high 0.083) on the normalized degradation scale $y^* \in [0, 1]$. This establishes that the two scales are largely monotone reparameterizations. *However*, a monotone mapping does not imply identical behavior under controlled severity shifts: coarse 5-point scores may reduce quantization jitter and improve within-slide ordering, whereas 100-point scores may expose finer variation that can either reflect genuine sensitivity or add noise. We therefore examine explicit *scale*×*dimension* trade-offs below.

Scale×dimension trade-offs. We quantify *internal consistency* as POA_{adj} and *fidelity* as $\text{Spearman}(\text{severity}, y^*)$. We find that for **geometry** and **style**, moving from 5-pt to 100-pt yields *no material fidelity gain* (bootstrap CIs overlap across models) but *reduces* POA_{adj}, as implied by the flat frontiers (*e.g.,* $[0.87, 0.95] \rightarrow [0.62, 0.73]$ (geometry); $[0.88, 0.98] \rightarrow [0.63, 0.81]$ (style)) (Fig. 3a). Thus a coarser scale is preferable for stability in these dimensions. In contrast, for **text**, 100-pt *increases fidelity substantially* (*e.g.,* GPT-5-high $0.51 \rightarrow 0.75$; GPT-5-minimal $0.52 \rightarrow 0.76$) while *lowering* POA_{adj} ($1.00 \rightarrow [0.88, 0.92]$), revealing a consistency-fidelity trade-off.

Model interchangeability. Models *diverge most on text* (Fig. 3b). Even the most convergent text pair (GPT-5-high vs. GPT-5-minimal) attains only $\overline{\rho} \approx 0.55$ (mean of per-severity Spearman), whereas geometry/style pairs frequently exceed $[0.80, 0.90]$. Notably, the most divergent geometry pair (*e.g.,* GPT-4o vs. o3) still shows higher agreement ($\overline{\rho} \approx 0.78$) than the most convergent text pair, underscoring that *text quality* is the most divergent axis for cross-model agreement.

Narrative in Slide Deck. Overall (Figure 8), the models exhibit difficulty in accurately predicting slide order, with Kendall's $\tau \in [0.04, 0.12]$, Spearman's $\rho \in [0.05, 0.13]$, and Exact Match scores $s \in [0.10, 0.17]$) only marginally outperforming random guessing, yet remaining below the theoretical upper bound of 1.0. This suggests that the models may struggle to comprehend and reason through the narrative flow of a presentation. Among them, GPT-4.1 delivered the strongest performance ($[0.04, 0.07]$ point of improvement) over GPT-5-minimal (Details in Appendix F.3).

5 Conclusion

We present **VLM-SlideEval**, a framework for evaluating slide element extraction, robustness to controlled perturbations, and narrative reordering on a curated PPTX corpus with ground truth. Newer VLMs (o3, GPT-5) outperform GPT-4.1/GPT-4o, yet all struggle with pixel-accurate style (*e.g.,* fonts) and cross-slide narrative coherence, and under perturbations exhibit a fidelity-consistency trade-off: geometry/style are comparatively stable, while finer text scales raise sensitivity but reduce internal score consistency. These findings argue for calibrated, slide-native evaluator in agentic/model-forward pipelines, using verifiable and accurate signals to gate selection and steer iterative refinement. Limitations include public PPTX, seeded perturbations, the suite of VLMs evaluated, as well as the simplified schema used for parsing slides; future work spans broader corpora, richer narrative probes, stronger verifiable checks, and judge calibration.

References

[1] Jiaxin Ge et al. "AutoPresent: Designing Structured Visuals from Scratch". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2025, pp. 2902–2911.

[2] Tsu-Jui Fu et al. "Doc2ppt: Automatic presentation slides generation from scientific documents". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 36. 1. 2022, pp. 634–642.

[3] Yang Xu et al. "LayoutLMv2: Multi-modal Pre-training for Visually-rich Document Understanding". In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics (ACL)*. 2021. URL: https://arxiv.org/abs/2012.14740.

[4] Dongsheng Wang et al. "DocLLM: A layout-aware generative language model for multimodal document understanding". In: *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*. 2024. URL: https://aclanthology.org/2024.acl-long.463/.

[5] Ray Smith. "An Overview of the Tesseract OCR Engine". In: *ICDAR '07: Proceedings of the Ninth International Conference on Document Analysis and Recognition*. Washington, DC, USA: IEEE Computer Society, 2007, pp. 629–633. ISBN: 0-7695-2822-8. URL: https://storage.googleapis.com/pub-tools-public-publication-data/pdf/33418.pdf.

[6] Steve Canny. *python-pptx: Create Open XML PowerPoint documents in Python*. https://github.com/scanny/python-pptx. Accessed: 2025-08-17. 2025.

[7] Souradip Chakraborty et al. "Review, Refine, Repeat: Understanding Iterative Decoding of AI Agents with Dynamic Evaluation and Selection". In: *arXiv preprint arXiv:2504.01931* (2025).

[8] Tianyi Xiong et al. "Llava-critic: Learning to evaluate multimodal models". In: *Proceedings of the Computer Vision and Pattern Recognition Conference*. 2025, pp. 13618–13628.

[9] Shuhang Liu et al. "MMC: Iterative Refinement of VLM Reasoning via MCTS-based Multimodal Critique". In: *arXiv preprint arXiv:2504.11009* (2025).

[10] Gio Paik, Geewook Kim, and Jinbae Im. "MMRefine: Unveiling the Obstacles to Robust Refinement in Multimodal Large Language Models". In: *arXiv preprint arXiv:2506.04688* (2025).

[11] Vidhisha Balachandran et al. "Eureka: Evaluating and understanding large foundation models". In: *arXiv preprint arXiv:2409.10566* (2024).

[12] Josselin S Roberts et al. "Image2struct: Benchmarking structure extraction for vision-language models". In: *Advances in Neural Information Processing Systems* 37 (2024), pp. 115058–115097.

[13] Viraj Prabhu et al. "Trust but verify: Programmatic vlm evaluation in the wild". In: *arXiv preprint arXiv:2410.13121* (2024).

[14] Yubo Ma et al. "Mmlongbench-doc: Benchmarking long-context document understanding with visualizations". In: *Advances in Neural Information Processing Systems* 37 (2024), pp. 95963–96010.

[15] Tianrui Guan et al. "Hallusionbench: an advanced diagnostic suite for entangled language hallucination and visual illusion in large vision-language models". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 14375–14385.

[16] An Vo et al. *Vision Language Models are Biased*. 2025. arXiv: 2505.23941 [cs.LG]. URL: https://arxiv.org/abs/2505.23941.

[17] Shuo Chen et al. "Benchmarking robustness of adaptation methods on pre-trained vision-language models". In: *Advances in Neural Information Processing Systems* 36 (2023), pp. 51758–51777.

[18] Tony Lee et al. "Vhelm: A holistic evaluation of vision language models". In: *Advances in Neural Information Processing Systems* 37 (2024), pp. 140632–140666.

[19] Kyudan Jung et al. "Talk to Your Slides: Language-Driven Agents for Efficient Slide Editing". In: *arXiv preprint arXiv:2505.11604* (2025).

[20] Kenton Lee et al. "Pix2struct: Screenshot parsing as pretraining for visual language understanding". In: *International Conference on Machine Learning*. PMLR. 2023, pp. 18893–18912.

[21] Yi-Hao Peng et al. "Slide gestalt: Automatic structure extraction in slide decks for non-visual access". In: *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. 2023, pp. 1–14.

[22] Jiaxin Ge et al. “Autopresent: Designing structured visuals from scratch”. In: *Proceedings of the Computer Vision and Pattern Recognition Conference*. 2025, pp. 2902–2911.

[23] Marco Lui and Timothy Baldwin. “langid.py: An Off-the-shelf Language Identification Tool”. In: *Proceedings of the ACL 2012 System Demonstrations*. Ed. by Min Zhang. Jeju Island, Korea: Association for Computational Linguistics, July 2012, pp. 25–30. URL: https://aclanthology.org/P12-3005/.

[24] Harold W Kuhn. “The Hungarian method for the assignment problem”. In: *Naval research logistics quarterly* 2.1-2 (1955), pp. 83–97.

[25] Russell Stewart, Mykhaylo Andriluka, and Andrew Y. Ng. “End-To-End People Detection in Crowded Scenes”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2016.

[26] Nicolas Carion et al. “End-to-end object detection with transformers”. In: *European conference on computer vision*. Springer. 2020, pp. 213–229.

[27] Yuyang Dong et al. “SCAN: Semantic Document Layout Analysis for Textual and Visual Retrieval-Augmented Generation”. In: *arXiv preprint arXiv:2505.14381* (2025).

[28] Baode Wang et al. “Infinity Parser: Layout Aware Reinforcement Learning for Scanned Document Parsing”. In: *arXiv preprint arXiv:2506.03197* (2025).

A Ground Truth Extraction Details

Ground truth elements are obtained by parsing the PowerPoint XML specification and cross-checking against a PNG export of the same slides. Each element type (text, rect, line, image, table) is represented in a unified schema with pixel-based geometry and absolute units for fonts and strokes (the full extraction schema is shown in Table 1 below).

Field(s)	Applies to	Unit / Notes
`w`, `h`	slide	px; fixed at 960×540
`x`, `y`, `w`, `h`	rect, text, image, table	px; top-left anchor
`x1`, `y1`, `x2`, `y2`	line	px; line endpoints
`rx`	rect	px; corner radius
`strokeWidth`	rect, line	points (pt); absolute width
`font.size`	text	pt; absolute font size
`font.style`	text	categorical; bold, italic, underscore
`color fields`	text, slide, line, rect	normalized hex (#RRGGBB)
`align`	text	categorical; left/center/right/justify/distributed

Table 1: Schema of extracted ground truth fields (excerpt). See Appendix A for full details.

We normalized the coordinates to the fixed slide size 960×540px, with its origin at the top-left corner. For styling information, font sizes are reported in points, while color values are normalized into #RRGGBB format. This enables precise cross-comparison between extracted ground truth and predictions returned by vision-language models (see Sec. 3). The summary statistics of ground truth element extraction can be found in Table 2

B Predicted Extraction Prompt

```
[System Message]
Analyze the location, size, and styling information of elements in the slide.
The size of the slide is: {TARGET_W} (w) x {TARGET_H} (h) pixels. The screenshot of the slide
      was taken at DPI = 72.
Top-left of the slide is (0,0), +x rightward, +y downward.
All geometry fields are integers in pixels, unless noted otherwise.

Return a JSON object with the following top-level fields for the single slide:
{ size, background, texts:[], rects:[], lines:[], images:[], tables:[] }.
Include every required field exactly as specified.
```

{*Extraction Specification Information: Table 1 Content Here*}

```
[User Message]
{"type": "image_url", "image_url": {"url": "<base64_thumbnail>", "detail": "auto"}}
```

Figure 4: Prompt used for structured extractions from VLMs for a single slide image.

We use a single-slide prompt that (i) fixes the slide coordinate frame at 960×540px with origin at the top-left; (ii) specifies units per field (pixels for geometry, points for fonts and strokes, hex for colors); and (iii) enumerates the required output schema (size, background, texts, rects, lines, images, tables) with field-level guidance (*e.g.,* x,y are the top-left of the element bbox; lines use x1,y1,x2,y2; rectangle corner radius is rx). The system message instructs the VLM to return a strict JSON object for the single image provided. A compact reference table in the prompt reiterates allowed values (*e.g.,* text align $\in$ {left, center, right, justify, distributed}) and clarifies that font and stroke widths are in points (absolute), while all positions and sizes are in pixels. The slide image is passed inline as a

	Per deck					Per slide					Total
Category	**Mean**	**SD**	**Min**	**Med**	**Max**	**Mean**	**SD**	**Min**	**Med**	**Max**	**Sum**
Num. of slides	19.48	11.54	1	18.0	46	—	—	—	—	—	1948
All elements	119.01	142.07	1	93.0	1183	6.11	9.03	0	4.0	153	11901
By type											
Text	63.40	58.40	0	49.0	314	3.25	3.34	0	3.0	69	6340
Rect	15.44	63.66	0	2.5	622	0.79	5.28	0	0.0	93	1544
Line	5.64	18.74	0	0.0	167	0.29	2.12	0	0.0	49	564
Image	33.71	33.50	0	28.0	172	1.73	2.54	0	1.0	44	3371
Table	0.82	4.09	0	0.0	40	0.04	0.35	0	0.0	11	82

Table 2: Ground-truth extraction summary across 100 decks and 1,948 slides. Per-deck statistics are computed across decks; per-slide statistics across slides.

base64 PNG. We enforce structured output via the API's JSON schema mode and validate responses with Pydantic; invalid JSON or schema mismatches are marked as parse failures.

Algorithm 1 Hungarian Matching with Blended Geometry+Content Cost and Threshold Gate

Input: $G = \{g_i\}_{i=1}^{m}$, $P = \{p_j\}_{j=1}^{n}$
Params: slide size (W, H); weights $(\alpha, \beta, \gamma, \delta)$; blended acceptance threshold $\tau \in [0, 1]$
Accessors: $\mathrm{box}(e) \to (x, y, w, h)$; $\mathrm{sim}(g, p) \in [0, 1]$ if available (else set δ=0)
Defs:
$\mathrm{IoU}(a, b) = \frac{\mathrm{area}(a \cap b)}{\mathrm{area}(a) + \mathrm{area}(b) - \mathrm{area}(a \cap b)}$
$d_{\mathrm{center}}(a, b) = \frac{\|c(a) - c(b)\|_2}{\sqrt{W^2 + H^2}}$ where $c(\cdot)$ is box center
$\mathrm{size\_rel}(a, b) = \frac{1}{2}\left(\frac{|w_a - w_b|}{\max(\varepsilon, w_a)} + \frac{|h_a - h_b|}{\max(\varepsilon, h_a)}\right)$
Construct $C \in \mathbb{R}^{m \times n}$
for $i = 1$ to m **do**
    for $j = 1$ to n **do**
        $a \leftarrow \mathrm{box}(g_i)$, $b \leftarrow \mathrm{box}(p_j)$
        $c_{\mathrm{iou}} \leftarrow 1 - \mathrm{IoU}(a, b)$; $c_{\mathrm{center}} \leftarrow d_{\mathrm{center}}(a, b)$; $c_{\mathrm{size}} \leftarrow \mathrm{size\_rel}(a, b)$
        $c_{\mathrm{cont}} \leftarrow 1 - \mathrm{sim}(g_i, p_j)$ **if** content available **else** 0
        $C_{ij} \leftarrow \alpha c_{\mathrm{iou}} + \beta c_{\mathrm{center}} + \gamma c_{\mathrm{size}} + \delta c_{\mathrm{cont}}$
    end for
end for
Compute optimal assignment $\mathcal{A} \subseteq \{1..m\} \times \{1..n\}$ by Hungarian on C
Threshold gate and bookkeeping
$\mathcal{M} \leftarrow \varnothing$; matchedG $\leftarrow \varnothing$; matchedP $\leftarrow \varnothing$
for each $(i, j) \in \mathcal{A}$ **do**
    if $C_{ij} \le \tau$ **then**
        $\mathcal{M} \leftarrow \mathcal{M} \cup \{(i, j)\}$; matchedG $\leftarrow$ matchedG $\cup \{i\}$; matchedP $\leftarrow$ matchedP $\cup \{j\}$
    end if
end for
Output: matches '$\mathcal{M}$', false positives '$P \setminus$ matchedP', false negatives '$G \setminus$ matchedG'

C Prediction-to-Ground Truth Matching Algorithm

Let $G = \{g_i\}$ denote the set of ground truth elements and $P = \{p_j\}$ the predicted elements. Each candidate match (g_i, p_j) ($c_{ij} \in C \in \mathbb{R}^{|G| \times |P|}$) we define a blended cost $c_{ij} = \alpha\big(1 - \mathrm{IoU}(g_i, p_j)\big) + \beta\, d_{\mathrm{center}}(g_i, p_j) + \gamma\, \mathrm{size\_rel}(g_i, p_j) + \delta\,\big(1 - \mathrm{sim}(g_i, p_j)\big)$, where IoU is the box overlap, d_{center} is normalized Euclidean center distance, $\mathrm{size\_rel}$ is relative size drift, and sim is a content similarity score (*e.g.,* normalized text similarity). We solve a minimum-cost bipartite matching with the

Hungarian algorithm [24, 26] on $C = [c_{ij}]$. Finally, we apply a lightweight sanity check: a matched pair (i, j) is accepted iff its blended cost is below a threshold τ (*i.e.,* $c_{ij} \leq \tau$); otherwise it is discarded, yielding an unmatched ground-truth (FN) and prediction (FP). Pseudo code of this procedure can be found in Algorithm 1.

This formulation generalizes naturally to other modalities; only the similarity term $\mathrm{sim}(\cdot)$ is type-dependent. For example, table elements may use cell-value overlap, and images may use caption, color histogram, and object-scene similarity.

D Perturbation Operators and Hyperparameters

Notation. We perturb a slide's element list $\mathcal{E}$ with a single strength knob $s \in [0, 1]$. When $s = 0$ the transform is a no-op (we return a deep copy). All probabilities and noise scales below are monotone in s, and all randomness is seeded for reproducibility.

Geometry (layout/alignment). We act on "box-like" elements with geometry (x, y, w, h) (text, image, table, rect, chart). For each eligible element (sampled with per-element probability π_{geo}; default $= 1.0$):

- **Translation:** $(x', y') = (x + \Delta_x,\ y + \Delta_y)$ with $\Delta_x \sim \mathcal{N}(0, \sigma_x^2)$, $\Delta_y \sim \mathcal{N}(0, \sigma_y^2)$,
$$\sigma_x(s) = (0.04 + 0.16\, s) \cdot W, \quad \sigma_y(s) = (0.04 + 0.16\, s) \cdot H,$$
where (W, H) is slide size (960×540px).
- **Scaling:** $(w', h') = (w \cdot \eta_w,\ h \cdot \eta_h)$, with $\eta_{\{\cdot\}} \sim \exp(\mathcal{N}(0, \sigma_{\log}^2))$ and $\sigma_{\log}(s) = 0.12 + 0.55\, s$.
- **Extreme size (optional):** with probability $p_{\mathrm{ext}}(s) = 0.20\, s$, additionally multiply (w', h') by
$$r \sim \mathrm{Uniform}(0.15, 0.50) \ \text{ or } \ \mathrm{Uniform}(1.5, 10).$$
- **Reposition (optional):** with probability $p_{\mathrm{rep}}(s) = 0.10\, s$, sample a fresh (x', y') uniformly over valid canvas positions (respecting current size).
- **Collapse (optional):** with probability $p_{\mathrm{col}}(s) = 0.08\, s$, set one dimension to $\mathrm{Uniform}(1, 3)$ px (skinny or flat).
- **Bounds:** clamp to $[0, W - w'] \times [0, H - h']$ unless `allow_clipping`.

Text Content. We operate on text elements; non-text are passed through. For each text box (sampled with per-element probability π_{txt}; default $= 1.0$):

- **Character-level noise** with per-character rate $p_{\mathrm{char}}(s) = p_{\min} + (p_{\max} - p_{\min})\, s$, where $p_{\min} = 0.02$, $p_{\max} = 0.25$. For each affected character, apply one of {substitute, delete, insert, adjacent-swap} with weights $(0.50, 0.20, 0.15, 0.15)$. Substitutions/insertions prefer keyboard-neighbor letters; case preserved.
- **Numeric preservation (optional):** after noise, restore the original numeric runs (`\d+(\.\d+)?`) in textual order to limit semantic drift on quantities.
- **Drop boxes (optional):** with probability $p_{\mathrm{drop}}(s) = 0.18\, s$, remove the entire text box.
- **Insert boxes (optional):** with probability $p_{\mathrm{ins}}(s) = 0.35\, s$, insert $n \in \{1, \ldots, \min(\texttt{max\_inserts},\ 1 + \lfloor 3s \rfloor)\}$ irrelevant text boxes. Each insertion samples geometry fractions $w/W \sim \mathrm{U}(0.15,\ 0.35 + 0.35s)$, $h/H \sim \mathrm{U}(0.08,\ 0.22 + 0.28s)$, with uniform valid (x, y). Text is drawn from a small pool (*e.g.,* "lorem ipsum", "TODO: revise"), and default font attributes are assigned (size scales with s; emphasis toggles with small s-scaled probabilities).

Style (typography & color). We act on text elements (per-element probability π_{sty}; default $= 1.0$). Let f denote a font object with fields {`name`, `size`, `bold`, `italic`, `underline`, `color`}.

- **Family switch:** with probability $p_{\mathrm{fam}}(s) = 0.20 + 0.60\, s$, replace `name` by a random choice from a fixed pool excluding the current family.
- **Size jitter:** $\texttt{size}' = \mathrm{clip}_{[6,120]}\big(\texttt{size} \cdot \exp(\mathcal{N}(0, \sigma_{\mathrm{sz}}^2))\big)$ with $\sigma_{\mathrm{sz}}(s) = 0.45\, s$. With probability $p_{\mathrm{szext}}(s) = 0.25\, s$, additionally multiply by $\mathrm{U}(0.12, 3.8)$ to produce tiny/huge outliers.
- **Emphasis toggles:** independently flip {`bold`, `italic`, `underline`} with probability $p_{\mathrm{tog}}(s) = 0.20\, s$.
- **Color:** with probability $p_{\mathrm{inj}}(s) = 0.30\, s$, inject an incongruent palette color (e.g., #FF0000, #FFFF00, #00FFFF, ...). Otherwise jitter the current color in HLS: $\Delta h \sim \mathrm{U}(-30°, 30°)\, s$,

$\Delta\ell \sim \mathrm{U}(-0.25, 0.25)\, s$, $\Delta s \sim \mathrm{U}(-0.20, 0.20)\, s$. With probability $p_{\mathrm{lowc}}(s) = 0.25\, s$, move toward the background color by $c' = (1-\alpha)c + \alpha\, c_{\mathrm{bg}}$ with $\alpha = 0.25 + 0.65\, s$.

- **Background:** with probability $p_{\mathrm{bg}}(s) = 0.20\, s$, jitter the slide background color as above.

E Additional Details for Analysis & Measures

E.1 Slide Parseability

Definition. A slide is counted as *parsed* if the model returns a JSON object that validates against our strict schema (fields, types, units) using Pydantic. Responses that are not valid JSON or violate the schema are marked as failures. Parseability is independent of matching quality (later we report on both the end-to-end - including parse failure cases where they would count towards the denominators of the downstream performance metrics - as well as the parsed-only - excluding parse failure cases from analysis; see Fig. 2 and Fig. 7 for the relevant results).

Complexity. We use GT scene complexity c as the total number of ground truth elements on a slide (sum over text, image, table, line, rect, table).

Reliability curve by complexity. Let $\{B_k\}$ be K quantile bins of c. For each bin B_k we report

$$\widehat{\Pr}(\text{success} \mid c \in B_k) \;=\; \frac{1}{|B_k|} \sum_{i \in B_k} \mathbb{1}\{\text{parsed}_i\},$$

with a 95% bootstrap confidence interval via percentile or BCa intervals.

E.2 Metric Definitions

To investigate the VLM slide comprehension accuracy, we measure a suite of metrics encompassing a diverse set of elements for the three dimensions of quality, as detailed below.

Matching counts & PRF1. For each family and overall (micro), precision $\mathrm{P} = \frac{TP}{TP+FP}$, recall $\mathrm{R} = \frac{TP}{TP+FN}$, and $\mathrm{F1} = \frac{2\mathrm{PR}}{\mathrm{P}+\mathrm{R}}$.

Geometry terms (interpretable). For boxes we report: $1 - \mathrm{IoU}$; center distance d^{center}; relative size r^{size}; for images, aspect-ratio error r^{ar}; for rectangles, radius error r^{rx}; for lines, relative length error r^{len} and angular error r^{ang}. All terms are in $[0, 1]$ after normalization. Lower is better.

Content similarity. Text strings are normalized by lowercasing, replacing "&→and", stripping punctuation, and collapsing whitespace. We compute $s^{\mathrm{content}} = \texttt{SequenceMatcher}(\tilde{t}_{\mathrm{pred}}, \tilde{t}_{\mathrm{gt}}) \in [0, 1]$ and also report $1 - s^{\mathrm{content}}$ where an error term is desired. (Embedding-based similarity is possible but not used in our primary results.)

Style. We measure color differences using CIEDE2000 (ΔE_{00}) computed in CIE $L^*a^*b^*$ space after sRGB→Lab conversion (D65; $k_L = k_C = k_H = 1$). Lower is better. *Rule-of-thumb*: $\Delta E_{00} \lesssim 0.5$ imperceptible, $0.5-1$ barely perceptible, $1-2$ small but visible, $2-3.5$ clearly noticeable under typical viewing. We evaluate: (i) slide background vs. GT; (ii) per element type—font color (`text`), fill and stroke (`rect`), and stroke (`line`). For numeric style fields we report absolute errors in native units: font size (pt) and stroke width (pt). For booleans we report mismatch rates (0/1): bold, italic, underline (for `text`). All statistics are summarized overall and per type using means, standard deviations, and counts; micro-averaged PRF1 is computed from summed TP/FP/FN.

Aggregation. We micro-average PRF1 by summing TP/FP/FN over all slides and runs. For scalar errors we report {mean, stdev, n} over all matched pairs (overall and by type). Where noted, we compute bootstrap 95% CIs (2,000 resamples). Deck-level summaries aggregate per slide, then pool across decks (pooled mean/stdev with sample-size weights).

Units & coordinate frame. All geometry is in a fixed 960×540px slide frame, with stroke width and font size in points. The rasterization is for screenshots only and does not alter the target coordinate system.

```
[System Message]
[Role]
You score the {DIMENSION} of a PowerPoint slide.

[Scale]
Return ONE integer on the scale {SCALE_MIN}..{SCALE_MAX} (inclusive).
Anchors:
- Min ({SCALE_MIN}): "{LOW_LABEL}".
{OPTIONAL_MID}- Mid ({SCALE_MID}): "{MID_LABEL}".
- Max ({SCALE_MAX}): "{HIGH_LABEL}".

[How to judge]
Consider only:
{CRITERIA_BULLETS}
```

```
[User Message]
{"type": "image_url", "image_url": {"url": "<base64_thumbnail>", "detail": "auto"}}
```

Figure 5: Prompt template used by VLM evaluators on perturbed slides.

E.3 Evaluator Prompts

The prompts used by VLMs for assessing the quality of perturbed slides along text, geometry, and style dimensions are instantiated using a common prompt template (Fig. 5). For *dimension*, we use {"text quality", "layout geometry", "style"}; we provide two scale set points {(1,5), (1,100)} and corresponding *mid-point* as the mean of the end-points, and labels as {"very poor", "acceptable", "excellent"}. The *how to judge* constraints in each dimension are shown in Table 3 below:

Text quality	**Layout geometry**	**Style**
• Clarity and plain language • Grammar/spelling • Bullet length (prefer one line) • Concision (avoid fluff)	• Alignment to grid/edges/baselines • Consistent spacing and margins • Balance and visual hierarchy • Element sizing matches importance	• Font family consistency and readability • Font size appropriate for viewing distance • Contrast and color harmony • Consistent emphasis (bold/italic/underline sparingly)

Table 3: *How to judge* constraints used by evaluators.

F Detailed Results

F.1 Slide Parsing Success Rate Conditioned on Scene Complexity

Parseability vs. complexity. Figure 6 visualizes trends across complexity bins; the per-bin summaries are:

- **GPT-5-high** is essentially at ceiling across all complexity bins: five bins are at $\mathbf{100}\%$ and the remaining two are $\mathbf{99.8}\%$–$\mathbf{99.9}\%$.
- **GPT-5-minimal** is likewise near-ceiling: $\mathbf{99.7}\%$–$\mathbf{100}\%$ in all but one bin; the lowest bin is $\mathbf{99.5}\%$ (16–32).
- **o3** remains at or near ceiling throughout, with $\mathbf{99.7}\%$–$\mathbf{100}\%$ across all bins.
- **GPT-4.1** shows clear sensitivity to complexity: $\mathbf{95.5}\%$ (0–1), $\mathbf{93.7}\%$ (1–2), $\mathbf{92.8}\%$ (2–4), $\mathbf{91.6}\%$ (4–8), then drops to $\mathbf{72.1}\%$ (8–16), $\mathbf{32.8}\%$ (16–32), and $\mathbf{18.2}\%$ (32–∞).
- **GPT-4o** underperforms `GPT-4.1` in most bins as complexity grows: $\mathbf{92.7}\%$ $(0, 1]$ $\mathbf{95.4}\%$ $(1, 2]$, $\mathbf{89.1}\%$ $(2, 4]$, $\mathbf{81.4}\%$ $(4, 8]$, $\mathbf{57.6}\%$ $(8, 16]$, $\mathbf{45.8}\%$ $(16, 32]$; the uptick to $\mathbf{66.7}\%$ in $(32, \infty]$ reflects small-sample volatility ($N = 66$).

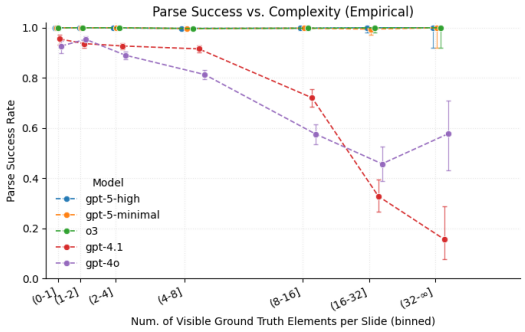


Figure 6: Parse success versus scene complexity (elements per slide) across VLMs. Complexity bins: $(0, 1]$, $(1, 2]$, $(2, 4]$, $(4, 8]$, $(8, 16]$, $(16, 32]$, $(32, \infty]$. GPT-5 and o3 remain near ceiling across bins, while GPT-4 series degrades with complexity. Estimates in the rightmost bin use small samples ($N = 66$ per model).

Small sample sizes in the extreme tail $(32, \infty]$, $N = 66$ per model) limit certainty there; the overall pattern is near-perfect parseability for the GPT-5 and o3 models, with sharp degradation for the GPT-4 series as complexity increases.

F.2 Extraction Performance

Fig. 7 summarizes extraction accuracy and geometry error with *Parsed Only* vs. *End-to-end* bars and coverage lines; Table 4 lists per-model metrics, showing e2e (parsed-only) in each cell with best e2e bolded. Overall, o3 and GPT-5-{minimal,high} lead across F1/accuracy and geometry, while GPT-4.1/GPT-4o degrade more under e2e, consistent with lower coverage.

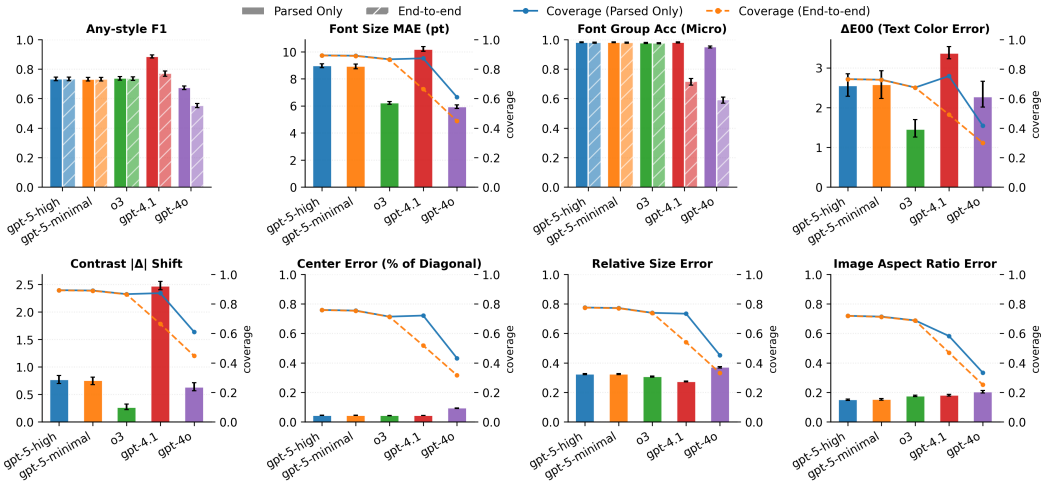


Figure 7: Bars show *Parsed Only* (solid) vs. *End-to-end* (hatched); lines (right axis) show **coverage** (fraction of ground-truth instances evaluated per metric). **Styling** (higher is better): Any-style F1 is moderate overall, with GPT-4.1 at 0.77 (best) and GPT-4o at 0.55 (worst); parsed-only boosts are pronounced for the 4-series (*e.g.,* 0.89 for GPT-4.1, 0.67 for GPT-4o). **Fonts**: font *group* accuracy is near-perfect for GPT-5-{minimal,high} and o3 (≥ 0.98) but lower for GPT-4.1/GPT-4o ($\approx 0.72/0.59$); font *family* accuracy is substantially lower across models $[0.17, 0.42]$. **Font size**: MAE (pt; lower is better) ranges $[5.93, 10.18]$ with GPT-4o best. **Color** (lower is better): text ΔE_{00} spans $[1.46, 3.37]$ (o3 best, GPT-4.1 worst) and contrast $|\Delta|$ shift spans $[0.26, 2.47]$ (o3 best, GPT-4.1 worst). **Geometry** (lower is better): $1 - \text{IoU}$ is best for o3 (0.55) and worst for GPT-4o (0.65); center error is $[0.04, 0.09]$, size error $[0.27, 0.37]$, and image aspect-ratio error $[0.15, 0.20]$. End-to-end coverage is substantially lower for the 4-series than for o3/GPT-5.

F.3 Slide Deck Narrative Order Performance

To assess narrative comprehension, we examine how effectively the VLM reconstructs the original sequence of slides from a randomly shuffled deck (Figure 8). Each deck is segmented into individual slide representations, which are then randomly reordered and input into the model along with a prompt instructing it to restore the correct order. The model's predicted sequence is evaluated against

Metric	GPT-4o	GPT-4.1	o3	GPT-5-minimal	GPT-5-high
Element Matching F1	0.44 (0.54)	0.59 (0.71)	**0.72** (0.72)	0.71 (0.71)	0.72 (0.72)
Geometry (micro; lower is better)					
1 − IoU	0.65	0.57	**0.55**	0.56	0.56
Center error (% diag)	0.09	0.04	**0.04**	0.04	0.04
Size error (relative)	0.37	**0.27**	0.31	0.32	0.32
Image AR error	0.20	0.18	0.18	0.15	**0.15**
Content (micro; higher is better)					
Text Content F1	0.63 (0.69)	0.69 (0.73)	**0.78** (0.78)	0.76 (0.76)	0.76 (0.76)
Style (micro; higher is better for style F1 and font accuracies; lower is better for color shifts)					
Any-style F1	0.55 (0.67)	**0.77** (0.89)	0.74 (0.74)	0.73 (0.73)	0.73 (0.73)
Font Family Acc (micro)	0.17 (0.27)	0.33 (0.45)	0.32 (0.32)	0.41 (0.41)	**0.42** (0.42)
Font Group Acc (micro)	0.59 (0.95)	0.72 (0.98)	0.98 (0.98)	0.98 (0.98)	**0.98** (0.98)
Font size MAE (pt)	**5.93**	10.18	6.22	8.92	8.97
Text color ΔE_{00}	2.27	3.37	**1.46**	2.57	2.55
Contrast $\lvert\Delta\rvert$ shift	0.63	2.47	**0.26**	0.75	0.77

Table 4: Extraction accuracy and geometry quality by model. Each cell shows *end-to-end* and (parsed-only) values, when applicable. Higher is better for F1/accuracy; lower is better for error metrics. Best model metric is boldfaced.

the ground truth using Kendall's τ, Spearman's ρ, and normalized exact match metrics. We report the mean and standard deviation across all decks.

As a preliminary step, we verify whether the models can generate output sequences that match the full length of the original presentations. For instance, if a presentation contains 23 slides, the model should produce an ordered list of 23 elements. According to Figure 6 (left), `GPT-5 high` and `o3` successfully generate nearly complete sequences, whereas other models struggle to even identify the correct number of slides present in the input.

Focusing on presentations with correctly predicted lengths, GPT-5-minimal and GPT-4.1 demonstrate relatively strong performance in ordering accuracy, as measured by Kendall's τ and Spearman's ρ, particularly outperforming `o3`. However, across the board, all models exhibit limited capability in narrative ordering, with scores below 0.15. This indicates substantial room for improvement before approaching the theoretical upper bound of 1.0 across all metrics. While the models appear capable of interpreting slide content and multimodal layout, they still face significant challenges in reasoning through the narrative structure.

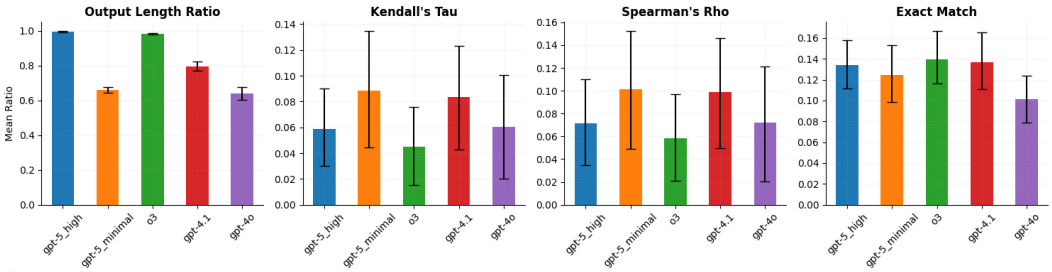


Figure 8: **Slide Deck Ordering Prediction**: 1) Output Length Ratio: GPT-5-high and o3 successfully generate nearly complete sequences 2) Kendall's τ and 3) Spearman's ρ: despite overlapping confidence integrals, GPT-5-minimal and GPT-4.1 show a consistent upward trend among these two measure, indicating potential robustness that warrants further investigation 4) Exact Match: models exhibit similar performance around 0.14 with GPT-4o being the lowest.

G Fonts and Font Groups Used in the Analysis

G.1 Canonicalized Font Names and Counts in the Dataset

Table 5 shows the count statistics of different fonts in text elements present in the ground truth slides.

Font	Count	Font	Count	Font	Count
calibri	2183	arial	1692	unknown	460
lato	260	montserrat	203	roboto	159
open sans	132	century gothic	105	oswald	105
helvetica neue	98	avenir next	97	garamond	70
verdana	66	ibm plex sans	65	corbel	64
georgia	61	source sans pro	53	libre franklin	43
tahoma	41	patrick hand	33	raleway	32
soehne	31	dosis	30	inter	22
times new roman	22	quattrocento sans	20	titillium web	20
bahnschrift	16	barlow	16	cambria	16
elephant	15	franklin gothic	14	nunito	14
gill sans	12	amatic sc	10	american typewriter	10
source code pro	10	ubuntu	9	ibm plex mono	5
palatino linotype	4	aptos	3	handwriting	3
segoe script	3	bookman old style	2	menlo	2
playfair display	2	tenorite	2	bodoni	1
inconsolata	1	pacifico	1	proxima nova	1
segoe ui	1	Total	6340		

Table 5: Frequency of different font families in the ground truth data (sorted descending, row-major)

G.2 Font → Font Group Mapping

```
# Sans
"arial":"sans","calibri":"sans","helvetica":"sans","helvetica neue":"sans","segoe ui":"sans","verdana":"sans",
"tahoma":"sans","gill sans":"sans","inter":"sans","roboto":"sans","open sans":"sans","lato":"sans",
"montserrat":"sans","source sans pro":"sans","libre franklin":"sans","quattrocento sans":"sans",
"ubuntu":"sans","barlow":"sans","bahnschrift":"sans","ibm plex sans":"sans","soehne":"sans","dosis":"sans",
"poppins":"sans","raleway":"sans","titillium web":"sans","nunito":"sans","corbel":"sans","candara":"sans",
"century gothic":"sans","avenir":"sans","avenir next":"sans","franklin gothic":"sans","arial rounded mt":"sans",
# Serif
"times new roman":"serif","georgia":"serif","garamond":"serif","cambria":"serif","palatino linotype":"serif",
"bookman old style":"serif","elephant":"serif","merriweather":"serif","playfair display":"serif",
"bodoni":"serif","bodoni mt":"serif","didot":"serif","tinos":"serif","cmr10":"serif","american typewriter":"serif",
# Mono
"courier new":"mono","courier":"mono","consolas":"mono","menlo":"mono","monaco":"mono","inconsolata":"mono",
"fira mono":"mono","source code pro":"mono","roboto mono":"mono","ibm plex mono":"mono",
# Script / Hand / Display
"comic sans ms":"script","brush script mt":"script","brush script":"script","amatic sc":"script",
"patrick hand":"script","architects daughter":"script","caveat":"script","pacifico":"script","lobster":"script",
"impact":"display","bebas":"display",
# Others
"roboto slab":"serif","carlito":"sans","asana":"serif","tenorite":"sans","aptos":"sans",
"segoe ui emoji":"sans","segoe ui symbol":"sans",
```

G.3 Font Group Frequencies

Table 6 shows the count statistics of different fonts in text elements present in the ground truth slides.

Font	Count	Font	Count	Font	Count
sans	5503	other	569	serif	203
script	47	mono	18	Total	6340

Table 6: Frequency of different font groups in the ground truth data (sorted descending, row-major)

H Reproducibility and Safety Checks for Slide Perturbation

- **Seeding:** All RNG draws use a fixed base seed; per-slide streams can be derived via a deterministic hash of the slide ID.
- **Validity:** Geometry is clamped to the canvas (unless explicitly allowed); sizes are lower-bounded by 1 px. Colors are validated to normalized hex (#RRGGBB) before export.
- **No-op at $s = 0$:** We return an unchanged copy when $s \leq 10^{-12}$.
- **On Monotonicity:** Because operations are stochastic, a single draw at $s = 1.0$ need not strictly dominate a draw at $s < 1$, but it does so at expectation (all scales/probabilities are monotone in s).

I Declaration of LLM Usage

We used large language model (LLM) assistants solely for *writing and tooling support*, including (i) manuscript/LaTeX editing, phrasing, and formatting, and (ii) non-substantive code assistance in VS Code (*e.g.,* refactoring, bug fixing, style cleanups, and commenting). All algorithms, evaluation designs, datasets, metrics, and reported results were specified by the authors; LLM-suggested text/code was reviewed, verified, and tested by the authors before inclusion. This usage does not impact the core methodology or conclusions.

Acknowledgments and Disclosure of Funding

We thank Naiqing Guan, Saket Gurukar, Md Muksitul Haque, Jinghua Yao, Lixiang Li, Ruolin Su, Sharena Pari-Monasch, Jesse Harvey, Dan Swett, and the anonymous NeurIPS reviewers for their constructive feedback.