DeepSlide: From Artifacts to Presentation Delivery

Ming Yang[*], **Zhiwei Zhang, Jiahang Li, Haoseng Liu, Yuzheng Cai, Weiguo Zheng**[†]
School of Data Science, Fudan University, Shanghai, China
{yangm24, zwzhang25, jiahangli25, hsliu23, yuzhengcai21}@m.fudan.edu.cn
zhengweiguo@fudan.edu.cn

GitHub: https://github.com/PUITAR/DeepSlide

Abstract

Presentations are a primary medium for scholarly communication, yet most AI slide generators optimize the *artifact* (a visually plausible deck) while under-optimizing the *delivery process* (pacing, narrative, and presentation preparation). We present *DeepSlide*, a human-in-the-loop multi-agent system that supports preparing the *full presentation process*, from requirement elicitation and time-budgeted narrative planning, to evidence-grounded slide–script generation, attention augmentation, and rehearsal support. DeepSlide integrates (i) a controllable logical-chain planner with per-node time budgets, (ii) a lightweight content-tree retriever for grounding, (iii) Markov-style sequential rendering with style inheritance, and (iv) sandboxed execution with minimal repair to ensure renderability. We further introduce a dual-scoreboard benchmark that cleanly separates static artifact quality from dynamic delivery excellence. Across 20 domains and diverse audience profiles, *DeepSlide* matches strong baselines on artifact quality while consistently achieving larger gains on delivery metrics, improving narrative flow, pacing precision, and slide–script synergy with clearer attention guidance.

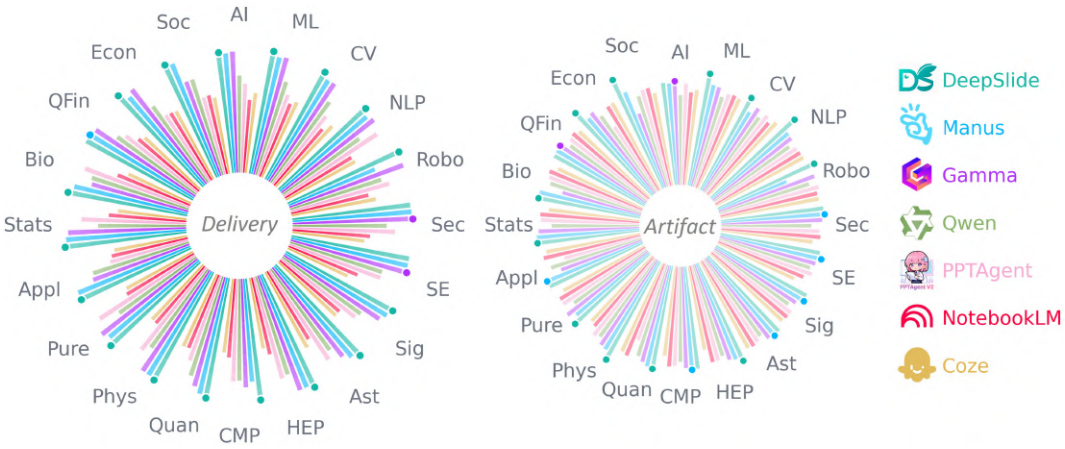


Figure 1: Main experiment on 20 domains (left is delivery scoreboard, while right is artifact scoreboard).

1 Introduction

Presentations are a primary medium for communicating information and ideas, playing a central role in scholarly exchange [1]. Delivering a high-quality talk typically requires not only a well-designed slide deck, but also a coherent, well-paced narrative aligned with the slides and substantial preparation (e.g., rehearsal and planning for audience interaction) [2]. Thus, talk preparation remains time-consuming, spanning both content authoring and delivery readiness.

[*]Project Leader
[†]Corresponding Author

Preprint.

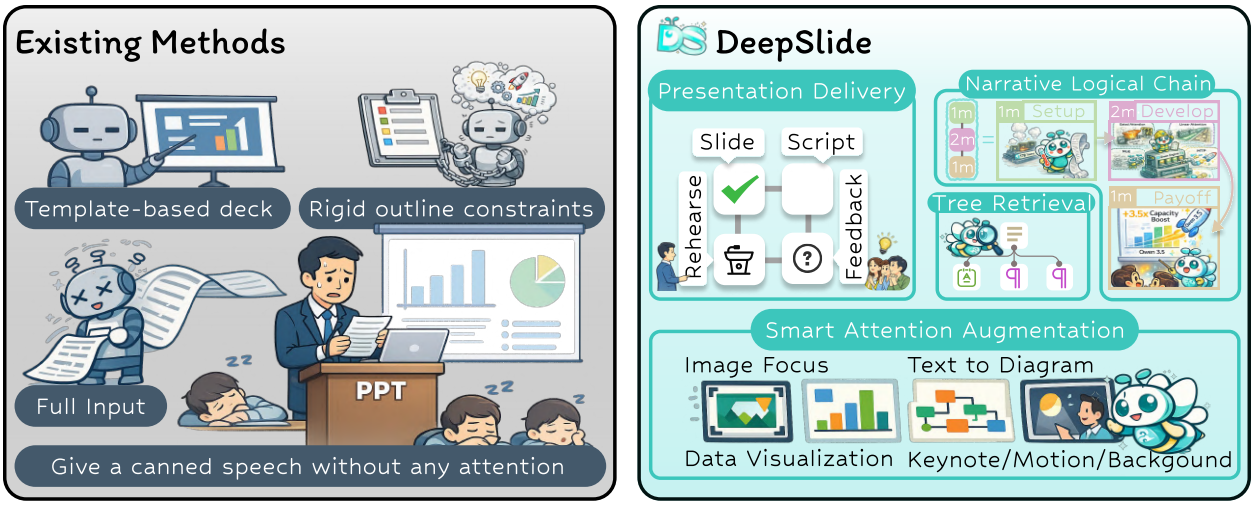


Figure 2: Limitations of existing approaches and the *DeepSlide*.

Recent advances in large language models (LLMs) and multimodal models have begun to reduce this burden [3–5] by improving document understanding and information extraction, and enabling increasingly reliable multimodal synthesis. Leveraging these model capabilities, agentic systems can ingest source materials, synthesize textual and visual content, and orchestrate multi-step workflows with structured decisions (e.g., outlining, emphasis, pacing). Meanwhile, engineering frameworks such as LangChain [6], AutoGPT [7], AutoGen [8], CAMEL [9], and MetaGPT [10] further lower the implementation cost of tool use and multi-agent coordination. Together, these advances have made presentation agents practical and productizable, exemplified by Manus [11], Gamma [12], NotebookLM [13], Qwen [14], Coze [15], and the open-source PPTAgent [16].These systems can already produce visually polished slide decks quickly, often with editing and refinement support.

However, prior research (e.g., [17]) together with practical observations suggests that a good presentation should not be defined primarily by how visually appealing the materials are. Instead, it hinges on whether information is structured and delivered in a way that aligns with the audience's cognitive and attentional constraints—clear organization, restrained distractions, and sustained guidance of audience attention. Consequently, even if these systems can produce polished and editable decks, artifact quality alone is insufficient to be equated with a high-quality presentation [1]. Effective presentation delivery is a human-centered communication problem, not a slide-design problem alone.

We summarize three key gaps in existing slide agents: (1) **Missing narrative strategy choices.** Most systems either skip explicit narrative planning or output a single, generic outline that is often non-editable and weakly personalized. More critically, they rarely treat *storytelling strategy* as a controllable design space: users are not offered multiple narrative styles to choose from (e.g., skeptic-to-believer persuasion, myth-busting reframing, trade-off navigation, or detective-style ablation reveal), nor are they guided to allocate time-budgeted emphasis accordingly (what to elaborate vs. compress). (2) **Limited delivery-time attention strategy.** Existing agents predominantly deliver static slide decks; for long-form logical arguments, complex experimental results, or detail-dense figures, they often lack content-aware mechanisms to guide attention during delivery (e.g., progressive revelation, focus/zoom cues, and tailored visual encodings). Although some agents (e.g., Qwen [14], Coze [15]) generate mind maps or data visualizations, these outputs are often static and template-driven, providing limited *delivery-time* attention guidance tailored to the actual slide content; predefined animations or template-based highlights, when available, are rarely content-aware enough to adapt attention strategy to dense figures and complex results. (3) **Insufficient rehearsal support.** Most approaches stop at producing slides, offering limited support for rehearsal and delivery preparation (e.g., generating a well-aligned, non-redundant script, anticipating live contingencies, and providing practice-time feedback), leaving users to design narration, plan interaction, and rehearse on their own. Overall, current slide agents mainly reduce the cost of deck authoring, but do not fully alleviate the end-to-end workload of presentation preparation.

To address these gaps, we propose *DeepSlide*, a four-stage (Fig. 3) end-to-end presentation agent that optimizes for full presentation delivery rather than static materials alone. First, instead of producing a generic, fixed outline, it elicits audience-, time-, and goal-specific requirements through free-form dialogue and generates multiple time-budgeted narrative logical-chain candidates, enabling users to select and refine an appropriate storytelling strategy tailored to the target audience before slide generation. Second, it enables *narrative-node-level* logical-chain editing—users can reorder/insert/remove/modify nodes, adjust per-node time budgets, and add non-linear cross-references—so the speaker

explicitly controls the macro storyline and emphasis plan; each node can be expanded into one or more slides during generation. Once the chain is finalized, *DeepSlide* retrieves supporting evidence from the indexed source materials to ground subsequent slide generation for each narrative node. Third, to better sustain audience attention during delivery, *DeepSlide* augments static decks with content-aware, selectable attention-control tools (e.g., narrative-intent-driven figure zooming, table-to-visualization transformation, and text-to-diagram concretization), together with interactive refinement and one-click layout optimization. Fourth, it goes beyond deck delivery by co-generating a synchronized, non-redundant speaker script aligned with the narrative nodes, supporting audience-perspective rehearsal (with optional audio), and providing evaluation with actionable revision suggestions and slide-level question simulation.

Together, these designs reduce end-to-end preparation workload by letting the speaker lock in high-level decisions (narrative skeleton, pacing/emphasis, and audience-conditioned style intent) while *DeepSlide* handles evidence-grounded realization, delivery-time attention augmentation, and rehearsal-time feedback.

Current benchmarks [18, 16, 19–23] mainly evaluate the *slide artifact* alone, leaving delivery-oriented qualities under-measured. Accordingly, we extend conventional static deck evaluation with a *dual-scoreboard* benchmark consisting of an *Artifact Scoreboard* and a *Delivery Scoreboard*. The former aligns with prior practice and assesses the quality of generated slide materials, while the latter evaluates delivery-oriented qualities, including narrative coherence, pacing control, and audience-attention guidance.

Contributions. In summary, we make the following contributions:

- We present *DeepSlide*, a four-stage, human-in-the-loop multi-agent system for deliverable presentation preparation.
- We introduce a dual-scoreboard benchmark that disentangles artifact quality from delivery quality, enabling evaluation beyond static slides.
- We develop lightweight yet principled mechanisms for controllable and reliable generation, including tree-aware retrieval for grounding, time-budgeted logical-chain editing with pacing feedback, Markov-style sequential rendering with style inheritance, and browser-sandbox validation for robust rendering.
- Extensive evaluations across 20 research domains and 5 audience profiles show that *DeepSlide* matches strong baselines on the Artifact Scoreboard while substantially improving the Delivery Scoreboard, reducing both preparation overhead and live-speaker effort.

2 *DeepSlide*

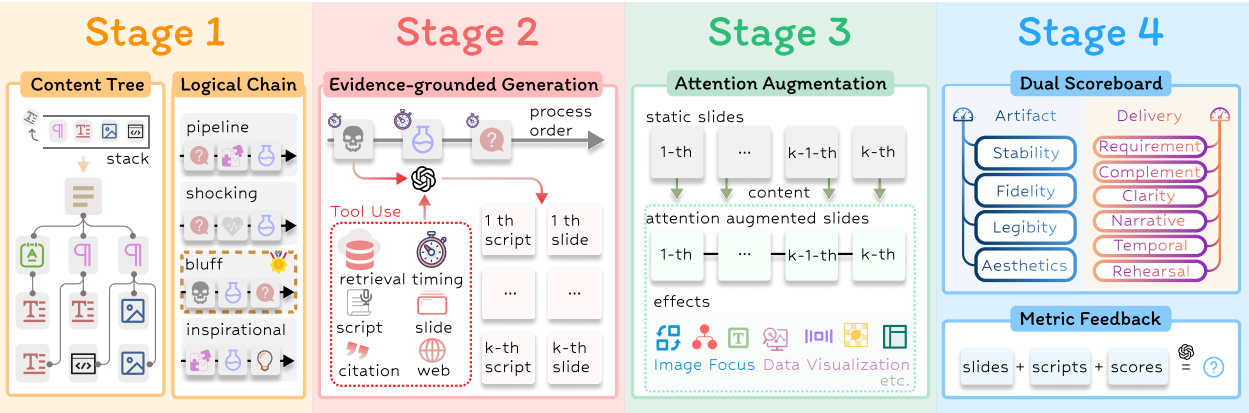


Figure 3: Overview. *Stage 1:* requirement elicitation and narrative proposal; *Stage 2:* logical chain editing and evidence-grounded generation; *Stage 3:* interactive slide refinement and attention-oriented augmentation; *Stage 4:* rehearsal and dual-scoreboard evaluation. Effects in Stage 3: *Image Focus*, *Text to Diagram*, *Keynote*, *Data Visualization*, *Motion*, *Background*, *Auto Layout* (Qwen3.5 [24] as example).

2.1 Overview

DeepSlide supports multi-source inputs in diverse formats and optimizes for the full delivery process. After materials are uploaded, it follows a four-stage pipeline.

Stage 1: Requirement elicitation and narrative proposal. *DeepSlide* conducts free dialogue (text or speech) to elicit presentation requirements, including the target audience, total duration, key emphasis, and preferred style, while building a lightweight *content-tree index* over the source materials for later evidence retrieval. It then produces four time-budgeted *narrative logic chain* candidates, each distributing the total duration in a well-balanced manner.

Stage 2: Logical chain editing and evidence-grounded generation. A logical chain is an ordered sequence of narrative nodes that abstracts the talk progression. DeepSlide supports logical chain editing (reorder/insert/remove/modify nodes, and adjust duration) and *non-linear narrative enhancement* by creating cross-references to serve as a connecting link between the preceding and the following. Once finalized, the system follows the chain, retrieves supporting evidence blocks from the content-tree index, and generates an audience-friendly, time-consistent slide along with a synchronized script aligned, where the two are designed to stay on-topic while minimizing redundancy.

Stage 3: Interactive slide refinement and attention-oriented augmentation. Users can refine slides and scripts via dialogue or speech with instant preview. *DeepSlide* provides one-click layout/typography optimization and mutiple freely selectable AI tools for delivery-time *attention control*: It identifies corresponding regions in complex figures based on the narrative intent and automatically zooms in on the corresponding areas. converts static tables into interactive visualizations; and turns verbose text into diagrams to improve readability and reduce cognitive load. It also supports text keynote, animations, and audio preview by extracting a speaker voice from conversational speech and synthesizing narration via TTS model [25].

Stage 4: Rehearsal and dual-scoreboard evaluation. *DeepSlide* supports audience-perspective rehearsal (with optional rehearsal audio). It further provides a dual-scoreboard evaluation that assesses artifact quality and delivery quality, and computes scores and revision suggestions grounded in the evaluation results, source materials, and user requirements. To further reduce preparation burden, it can simulate likely audience questions. Finally, DeepSlide enables one-click packaging/export.

2.2 Stage 1: Requirement Elicitation and Narrative Proposal

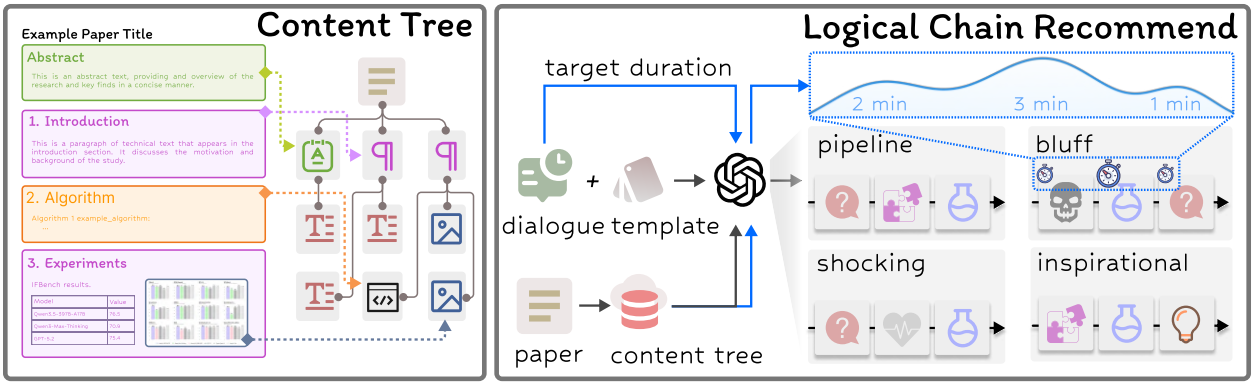


Figure 4: Requirement elicitation and narrative proposal (Stage 1).

This stage aligns the user's presentation intent with the paper content and, under a time budget, proposes candidate narrative modes as a stable scaffold for slide generation. Guided by system-prompt constraints, the agent elicits key requirements (audience, duration, focus, and style) and uses paper-aware follow-up questions to converge to a complete presentation blueprint through multi-turn dialogue. The dialogue yields two artifacts: a content tree and a set of candidate logical chains, which is shown in Fig 4.

Content tree. Naively passing the full paper to an LLM on every generation step is both token-wasteful and imprecise: large undifferentiated text blocks introduce irrelevant context that degrades generation quality. A vector-database pipeline avoids this but is unnecessarily heavy for single-document parsing. The content tree occupies the middle ground: a typed, hierarchical index built directly from the source, enabling targeted retrieval without embeddings or approximate-search infrastructure. Construction proceeds in three steps. ***(i) Source normalization.*** Mixed uploads (LaTeX, Markdown, Word, PDF, or nested archives) are unified into LaTeX via `pandoc`; the root file is located and all `\input`/`\include` directives are recursively flattened into a single stream. ***(ii) Lossless segmentation.*** Structural headings partition the stream into typed slices; artifact slices (`type`$\in\{$`equation`,`figure`,`table`,`theorem`,`algorithm`$\}$) are never merged

Algorithm 1 Hierarchical content tree construction.

Require: Ordered nodes $\mathcal{N} = \langle n_1, \ldots, n_m \rangle$ with integer level $\ell(n)$
Ensure: Root list $\mathcal{R}$ and parents/children for all nodes

$\mathcal{R} \leftarrow \emptyset$, $S \leftarrow \emptyset$ ▷ monotonic stack increasing levels
for $i \leftarrow 1$ to m **do**
  $n \leftarrow n_i$
  while $S \neq \emptyset \wedge \ell(\text{top}(S)) \geq \ell(n)$ **do** pop(S)
  if $S = \emptyset$ **then** append n to $\mathcal{R}$
  else $p \leftarrow \text{top}(S)$, $n.parent \leftarrow p$, $p.children \leftarrow p.children \cup \{n\}$
  push(S, n)
return $\mathcal{R}$

with adjacent text, preserving them as atomic retrieval units. Each slice is assigned an LLM-generated `abstract` summarizing its own content. ***(iii) Hierarchical index construction.*** Each slice becomes a `ContentTreeNode` $\langle id, title, level, type, abstract, content \rangle$, where *level* follows the document hierarchy (ℓ=1 for `\section`, ℓ=2 for `\subsection`, etc.). The slices are retained in document order, forming the flat sequence $\mathcal{N} = \langle n_1, \ldots, n_m \rangle$; a monotonic stack S (Alg. 1) then links them into a tree in a single $O(m)$ pass: when processing n_i, all nodes with $\ell(S.\text{top}) \geq \ell(n_i)$ are popped; the remaining top is n_i's parent (or n_i becomes a root if S is empty), then n_i is pushed. S thus always holds the unique ancestor chain from the current root to the deepest open section, correctly nesting each node under its nearest enclosing heading. The resulting tree mirrors the paper's logical outline and supports context-aware BM25 retrieval: each node's relevance score is jointly influenced by its own content, its children's scores, and its parent's score, enabling multi-granularity access from section-level overviews down to fine-grained leaf nodes.

Logical chain recommendation. After the content tree is built, DeepSlide generates four candidate logical chains, each embodying a distinct narrative style, via two complementary pipelines. *(i) Template-guided think–retrieve generation.* The agent first selects four narrative templates from a template library conditioned on the multi-turn dialogue history and the paper abstract. For each template, it executes a think–retrieve loop: the agent retrieves the paper outline and BM25-ranked evidence from the content tree, then synthesizes the retrieved content with the dialogue context and template constraints to produce a personalized chain. *(ii) Duration-aware time allocation.* The agent parses the target talk duration from the dialogue history and retrieves section-level evidence to estimate relative node importance. It then distributes the total time across nodes proportionally to their `importance ratios`.

2.3 Logical Chain Editing and Evidence-grounded Generation (Stage 2).

Stage 2 supports editing of the selected logical chain, and evidence-grounded slide generation (Fig 5).

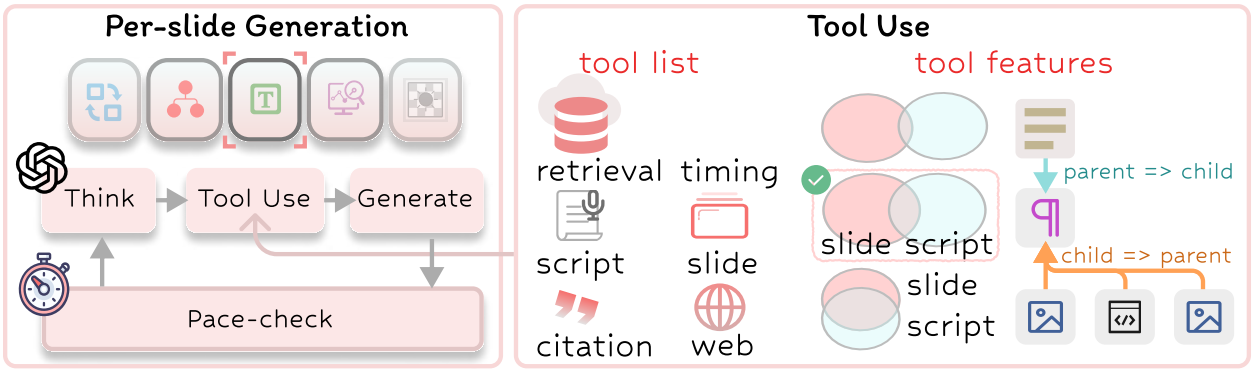


Figure 5: Logical chain editing and evidence-grounded generation (Stage 2).

2.3.1 Logical Chain Editing

Stage 2 enables users to refine the narrative plan before slide generation by editing the logical chain. Users can reorder, insert, remove, or modify logical nodes (e.g., "add an ablation slide" or "move limitations earlier"), and adjust per-node time budgets to match the target duration.

Algorithm 2 Tree-aware BM25 retrieval over a content tree.

Require: Nodes $\mathcal{N}$, corpus stats $\mathrm{df}/\mathrm{dl}/\mathrm{avgdl}$, fusion params $(\alpha_{\mathrm{tree}}, \beta_{\mathrm{tree}})$, depth-bias params $(\gamma_{\mathrm{tree}}, \delta_{\mathrm{tree}})$, threshold m_0, query q, top-K
Ensure: Ranked nodes R

$\mathcal{Q} \leftarrow \mathrm{tokenize}(q), \quad m \leftarrow |Q|$
for all $t \in \mathcal{Q}$ **do** $\mathrm{idf}(t) \leftarrow \log\Big(1 + \frac{|\mathcal{N}| - \mathrm{df}(t) + 0.5}{\mathrm{df}(t) + 0.5}\Big)$
for all $n \in \mathcal{N}$ **do** $s_0(n) \leftarrow \mathrm{BM25}(n, \mathcal{Q}, \mathrm{idf}, \mathrm{dl}, \mathrm{avgdl})$
for all $n \in \mathcal{N}$ **do** ▷ $s_0(\mathrm{parent}(n)) = 0$ if root
  $s(n) \leftarrow s_0(n) \; + \; \alpha_{\mathrm{tree}} \cdot \sum_{c \in \mathrm{children}(n)} s_0(c) \; + \; \beta_{\mathrm{tree}} \cdot s_0(\mathrm{parent}(n))$
  if $m \le m_0$ **then** $s(n) \leftarrow s(n) \cdot \frac{1}{1 + \gamma_{\mathrm{tree}} \cdot \mathrm{depth}(n)}$
  else $s(n) \leftarrow s(n) \cdot (1 + \delta_{\mathrm{tree}} \cdot \mathrm{depth}(n))$
return $\mathrm{topK}(\{(n, s(n)) : s(n) > 0\}, K)$

2.3.2 Content Tree Retrieval

Okapi-BM25 index. To support fast/precise retrieval, we overlay a lightweight Okapi-BM25 index on Stage 1 content tree. We strip LaTeX commands, truncate overly long spans by hyperparameter $l_{\max}$, and compute corpus statistics once, including document length $\mathrm{dl}(n)$, average length avgdl, and term document frequency $\mathrm{df}(t)$.

Tree-aware scoring. Given a query q, we first tokenize it into a multiset of terms $Q = \mathrm{tokenize}(q)$ and compute a base Okapi-BM25 score $s_0(n)$ for every node-document $n \in \mathcal{N}$. In our implementation, BM25 is computed with standard length normalization (Alg. 2): for each query term $t \in Q$, we form $\mathrm{idf}(t) = \log\big(1 + \frac{|\mathcal{N}| - \mathrm{df}(t) + 0.5}{\mathrm{df}(t) + 0.5}\big)$ and accumulate a saturated term-frequency contribution $s_0(n) = \sum_{t \in Q} \mathrm{idf}(t) \cdot \frac{\mathrm{tf}(t,n) \cdot (k_1 + 1)}{\mathrm{tf}(t,n) + k_1 \cdot (1 - b + b \cdot \mathrm{dl}(n)/\mathrm{avgdl})}$, where $\mathrm{tf}(t, n)$ is the within-node term frequency and $\mathrm{dl}(n)$ is the node-document length. We then inject structural priors from the content tree in two complementary directions. First, *child-to-parent promotion* makes a section-level node competitive when its descendants contain the most relevant evidence; second, *parent-to-child promotion* propagates topic-level relevance downward so that fine-grained nodes are not overly penalized when the query matches a broader heading. Concretely, we define the tree-aware score

$$s(n) = s_0(n) + \underbrace{\alpha_{\mathrm{tree}} \sum_{c \in \mathrm{children}(n)} s_0(c)}_{\text{children}\to\text{parent}} + \underbrace{\beta_{\mathrm{tree}} s_0(\mathrm{parent}(n))}_{\text{parent}\to\text{children}}. \tag{1}$$

with $s_0(\mathrm{parent}(n)) = 0$ for roots. Finally, we apply a query-granularity depth bias to align retrieval granularity with user intent: if $|\mathcal{Q}| \le m_0$ (overview queries), we downweight deep nodes by $\frac{1}{1 + \gamma_{\mathrm{tree}} \cdot \mathrm{depth}(n)}$; otherwise (detail queries), we upweight deep nodes by $(1 + \delta_{\mathrm{tree}} \cdot \mathrm{depth}(n))$. The total query-time cost is dominated by computing $s_0(n)$ over all nodes and aggregating child scores over edges. With precomputed term statistics, the scoring is $O(|\mathcal{N}| \cdot |\mathcal{Q}| + |\mathcal{E}|)$.

2.3.3 Evidence-grounded Generation

DeepSlide instantiates a tool-augmented generation agent to convert each logical-chain node into a short segment of slides and an aligned speaker script under a strict per-node time budget. Rather than a single-pass "retrieve once, then write" workflow (e.g., Gamma [12], NoteBookLM [13]), the agent follows a multi-turn loop: it iteratively retrieves evidence, drafts a candidate slide, checks pacing, and then decides whether to refine, expand, or stop. This closed-loop design is critical for producing delivery-ready artifacts that remain grounded while being sensitive to time and narrative constraints.

Tool-call driven multi-round reasoning. For a target logical node with a topic name and description, the agent is equipped with a small set of retrieval and creation tools, and can call them multiple times in a single generation of a logical node:

- *Tree retrieval and reading.* The agent first retrieves top-K relevant content-tree nodes w.r.t its thinking using the tree-aware scoring in Alg. 2. It then reads the full content spans of the selected nodes to collect grounded evidence before writing.

- *Complementarity between slide and script.* After interpreting the evidence, the agent can append (i) a new slide frame, (ii) its aligned speaker script, and (iii) optional bibliography entries. Slide content and spoken script are designed to be *highly related but not redundant*: slides keep key points and visual anchors, while details is delegated to the script.
- *Pacing control.* A countdown timer w.r.t. a logical node is continuously updated and, after each slide is generated, produces immediate feedback based on the estimated script length to regulate the remaining generation granularity under the time budget. When ample time remains, the timer stays silent; once the remaining time drops below half, it issues a cautionary prompt; when the budget is exhausted, it instructs the agent to aggressively compress and finish; and if a preset overrun threshold is exceeded, it forcibly terminates further slide generation.
- *Web search.* When a logical node requires background that is not present in the uploaded document, the system can optionally query the web to fetch supporting evidence.

2.3.4 Non-linear Narrative Enhancement and Compilation

Beyond linear ordering, DeepSlide also supports *non-linear narrative enhancement*: the agent may automatically create cross-references between nodes to bridge the preceding and following context, improving transitions and global coherence (e.g., foreshadowing an upcoming result or recall a prior definition). A compilation agent renders the slides under system prompts, automatically detects build failures, and performs sandboxed debugging in a Docker container via iterative read/write and code-execution tools.

2.4 Stage 3: Interactive slide refinement and attention-oriented augmentation

Stage 3 turns the static outputs from Stage 2 into a delivery-ready, interactive deck. Let the deck be a slide sequence $\{f_k^{\mathrm{src}}\}_{k=1}^{m}$. Stage 3 produces an augmented dynamic sequence $\{f_k^{\mathrm{dst}}\}_{k=1}^{m}$, while supporting refinement via text/voice dialogue (see Fig 6).

2.4.1 Dialogue Refinement

DeepSlide supports user-in-the-loop refinement at both the slide and script levels. A user refinement instruction can be given via text or speech (transcribed by ASR). We use a planner–executor manner to translate complex instruction into a few edit actions and apply minimal, localized updates to the affected parts.

2.4.2 Attention Augmentation

The core goal is to convert the source sequence $\{f_k^{\mathrm{src}}\}_{k=1}^{m}$ into an augmented sequence $\{f_k^{\mathrm{dst}}\}_{k=1}^{m}$ under two simultaneous optimization targets: *(i) stability/fidelity*, preserving the original content and structure, and *(ii) attention gain*, introducing controlled dynamic effects that improve audience attention and delivery clarity. Unlike template-based PPT agents that hard-code stability by forcing slides into fixed layouts, DeepSlide adopts a two-step augmentation framework. First, we augment slides in delivery order via Markov-style sequential control, where the agent triggers deterministic effect functions to improve focused attention under predefined strategies; Second, since template-free rendering may be unstable, we validate the generated deck in a sandboxed browser and feed back runtime errors to revise the plans, ensuring successful rendering finally. This encourages a consistent style while maximizing the dynamic/modern rendering of an agent-based template-free approach.

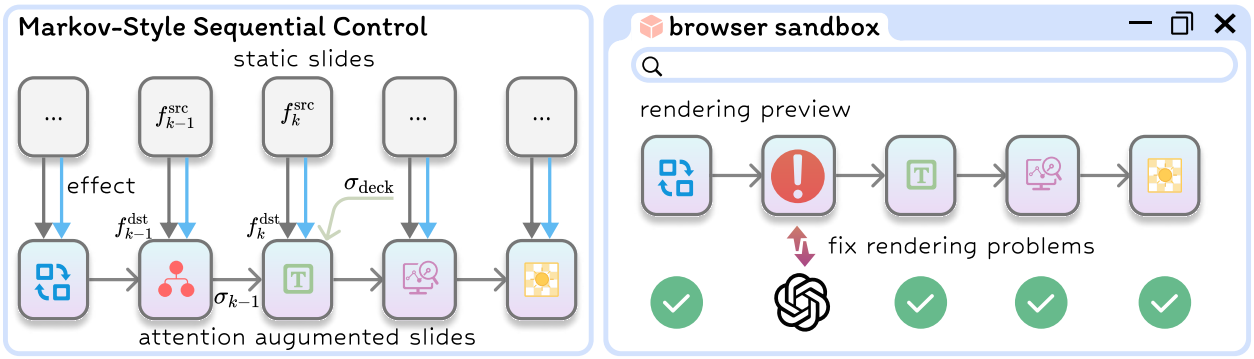


Figure 6: Interactive slide refinement and attention-oriented augmentation (Stage 3), where f_k^{src} and f_k^{dst} are the original and augmented slides, σ_{deck} is the Stage 1 deck style, and σ_{k-1} is the previous-slide style.

Algorithm 3 Markov-style sequential generation with style inheritance.

Require: Source slides $\{f_k^{\mathrm{src}}\}_{k=1}^m$, requirement profile $\mathcal{U}$, deck style summary σ_{deck}, planner π, renderer T, style summarizer S
Ensure: Augmented slides $\{f_k^{\mathrm{dst}}\}_{k=1}^m$
$\sigma_0 \leftarrow \sigma_{\mathrm{deck}}$
for $k \leftarrow 1, \ldots, m$ **do**
  $s_k \leftarrow \left(f_k^{\mathrm{src}}, \mathcal{U}, \sigma_{\mathrm{deck}}, \sigma_{k-1}\right), a_k \leftarrow \pi(s_k), f_k^{\mathrm{dst}} \leftarrow T(f_k^{\mathrm{src}}, a_k)$
  $\sigma_k \leftarrow$ summarize style description for next Markov state
return $\{f_k^{\mathrm{dst}}\}_{k=1}^m$

Markov sequential control. As shown in Fig 6, for slide f_k, an LLM-based planner conditions on its source content f_k^{src}, a user requirement profile $\mathcal{U}$, a deck-level style summary σ_{deck}, and a compact summary of the previously generated slide σ_{k-1}, yielding a Markov-style(first-order) dependency: $s_k = (f_k^{\mathrm{src}}, \mathcal{U}, \sigma_{\mathrm{deck}}, \sigma_{k-1}), \quad f_k^{\mathrm{dst}} = T\big(f_k^{\mathrm{src}}, \pi(s_k)\big)$, where π denotes the LLM that maps the state s_k to a structured augmentation decision for slide k (e.g., which effects to apply and how to stage them), and T is a deterministic renderer that executes the decision to produce f_k^{dst}. The overall process is shown in Alg 3, where the term σ_{k-1} enforces cross-slide style continuity (instead of any template), while f_k and $\mathcal{U}$ enable per-slide, content-conditioned customization under validity constraints.

Attention strategies. DeepSlide represents attention enhancement as freely selectable effect set $\mathcal{E}$ encoded in the Stage 3 and rendered deterministically. We separate effects into four classes:

- *Image focus.* This feature enables localized focus on complex figures during delivery. Rather than allowing arbitrary boxes (current VLMs struggle to identify focused regions), the system selects a focus template from a small predefined set (e.g., left/right split, 2×2 grid), which deterministically maps to normalized regions of interest (ROIs). The renderer then generates clickable ROI tiles with a lightbox interaction to guide attention without modifying the original figure.
- *Text to diagram.* For slides with convoluted and verbose text, we switch to a diagram-style layout to improve delivery clarity. We use a LLM to draft a diagram design specification from f_k^{src}'s text and then invoke an open-source diagram generator (e.g., next-ai-draw-io [26]) to render it.
- *Data Visualization.* When f_k^{src} contains tabular data, the system can detect it and generate a table-centric design, which is then rendered as interactive visualizations using open-source frontend libraries such as ECharts [27] with matching form.
- *Other Effects.* We further provide lightweight attention cues that add no new semantic content. For example, *Text Keynote* highlights a few critical phrases (e.g., key numbers "3×"). *Auto Layout* restructures dense paragraphs into clearer visual organizations (e.g., step-wise process blocks or card-style summaries). *Motion* introduces subtle entry animations to guide reading order, and *Background* non-intrusively adds a content-aware, *bento*-style backdrop that improves visual coherence without occluding the main text.

Effect conflict and gating. To prevent conflicts between effects, we apply a deterministic gating rule before planning: each slide can have *at most one primary visual effect* from {Image Focus, Table Viz, Text to Diagram}, with priority Image Focus $\succ$ Table Visualization $\succ$ Text to Diagram. The effect is selected if and only if the effect set $\mathcal{E}$ includes "structural recognition" (e.g., figure/tabular) and the priority conditions are satisfied.

Sandbox-guided rendering stabilization. Since template-free augmentation can occasionally render poorly, *DeepSlide* executes each generated slide in a lightweight browser sandbox to detect layout and runtime failures. Detected issues are fed back for a minimal repair.

2.4.3 Audio Rehearsal with User Voice.

DeepSlide provides a per-slide speech audio preview by synthesizing p_k into a waveform via TTS [25]. The user's voice profile is derived from both a audio library and real-time voiceprint captured during previous dialogues, which enables users to review their presentation from an audience's perspective.

2.5 Stage 4: Rehearsal and dual-scoreboard evaluation

Stage 4 turns a generated deck into a deliverable talk by providing rehearsal guidance (including audio preview generated in Stage 3) and post-hoc evaluation. The dual-scoreboard protocol is formalized in Section 3; here we focus

on how *DeepSlide* uses *slides + scripts + evaluation signals* to give concrete revision suggestions and to simulate likely audience questions.

For each slide, we assemble a compact context ⟨ `slide`, `script`, `per-slide metrics`, `estimated time comsume` ⟩ and prompt an LLM to produce: (i) 3–6 short rehearsal tips that are specific to the slide (e.g., where to slow down, what to cut, what to emphasize), and (ii) the top-3 likely audience questions for that slide, biased toward metrics-indicated risks (e.g., unclear assumptions, missing baselines, or abrupt topic jumps). These support an iterative rehearsal: speakers rehearse with the audio preview, inspect advice/questions, and revise slides/scripts before re-running the diagnostics.

Example 1. *Consider a slide titled* "Ablation Study" *without any DeepSlide's augmentation whose body contains a dense table and long bullet list, while its script repeats most on-slide text. DeepSlide estimates a* 75*s narration for a* 45*s time budget and detects high slide–script similarity, leading to the following coaching outputs:* ***(i) Suggestions:*** *"Remove two minor ablations"; "State one takeaway before details"; "Move table reading to appendix"; "Explain the strongest baseline first".* ***(ii) Likely audience questions:*** *"Which component contributes most to the gain?"; "Are improvements statistically significant?"; "Why is method X not included as a baseline?".*

2.6 Multi-Agents

DeepSlide employs a role-based multi-agent pipeline. Rather than unconstrained generation, each agent performs a bounded task, e.g., requirement normalization, logical planning, or rehearsal coaching, and updates shared structured artifacts. This decomposition enhances controllability, enabling stage-wise verification and localized repair. We treat each system-prompted role as a distinct agent, regardless of the underlying model. All agents in used are in Table 4.

3 Dual-Scoreboard Evaluation

We evaluate agents using a dual-scoreboard protocol that decouples static slide quality from end-to-end delivery performance. For each instance consisting of a source document $\mathcal{T}$ and requirement profile $\mathcal{U}$ (e.g., audience and duration), systems generate a package of slides $\mathcal{F} = \{f_k\}_{k=1}^{m}$ and aligned scripts $\mathcal{P} = \{p_k\}_{k=1}^{m}$, with $\tau = 5$ runs to ensure stability. The Artifact Scoreboard assesses the visual and content quality of the slides, while the Delivery Scoreboard measures narrative coherence and slide-script coordination, ensuring the system explains rather than merely duplicates on-slide text to meet audience-specific constraints.

3.1 Artifact Scoreboard

The Artifact Scoreboard evaluates systems across four dimensions: Stability, Fidelity, Legibility, and Aesthetics.

Stability. We measure stability by the success rate $P = \frac{\#\{r \in \{1,\dots,\tau\}:\, \text{run } r \text{ succeeds}\}}{\tau}$, where a run is counted as successful if it produces a usable deck with complete slide–script alignment.

Fidelity. We quantify how faithfully the generated presentation is grounded in the source. For each slide k, we align the slide–script pair (f_k, p_k) to the most relevant source chunk(s) in $\mathcal{T}$ via retrieval, and denote the resulting reference text by t_k. Textual fidelity F_t is a weighted combination of a lexical-overlap metric (ROUGE) and a semantic-similarity metric (BERTScore), averaged over both slide text and the aligned script:

$$F_t = \frac{\omega_{\text{rouge}}}{2m} \sum_{k=1}^{m} \Big(\text{ROUGE}(f_k, t_k) + \text{ROUGE}(p_k, t_k) \Big) + \frac{\omega_{\text{bert}}}{2m} \sum_{k=1}^{m} \Big(\text{BERT}(f_k, t_k) + \text{BERT}(p_k, t_k) \Big), \quad (2)$$

where ω_{rouge} and ω_{bert} control the contribution of the two metrics ($\omega_{\text{rouge}} + \omega_{\text{bert}} = 1$). Visual fidelity F_v is defined as the fraction of slides whose visual items are supported by the aligned source:

$$F_v = \frac{1}{m} \sum_{k=1}^{m} \mathbb{I}\left\{\text{slide } f_k \text{ contains a visual element matching } t_k\right\}, \quad (3)$$

where $\mathbb{I}\{\cdot\}$ is the indicator function that equals 1 if the condition is satisfied and 0 otherwise.

Legibility and aesthetics. We compute an automatic legibility score $L \in [0, 1]$ from slide-level readability checks. For each slide f_k, we extract (i) the minimum fontsize, (ii) the number of words in on-slide text. We define a per-slide $\text{penalty}_k = \mathbb{I}[\text{minfont}(f_k) < 12] + \mathbb{I}[\text{wc}(f_k) > 140]$, and aggregate the legibility score

$$L = 1 - \frac{1}{m} \sum_{k=1}^{m} \min(1, \text{penalty}_k), \quad (4)$$

Since purely subjective aesthetics is hard to standardize, we use an automatic Vision-Language Model (VLM) generate $Ae \in [0,1]$ that favors readable decks with balanced content. We define the aesthetic score as

$$Ae = \mathrm{clip}\Big(0, 1,\ 0.6L + 0.2\Big(1 - \big|\mathrm{frac}_{\mathrm{img}} - 0.6\big|\Big) + 0.2\,\mathrm{frac}_{\mathrm{script}}\Big), \tag{5}$$

where $\mathrm{frac}_{\mathrm{img}}, \mathrm{frac}_{\mathrm{script}} \in [0,1]$ are the fraction of slides that contain at least one image and the fraction of slides with speaker scripts. The weights of 0.6 and 0.2 are empirical hyperparameters.

Aggregation. We report the Artifact score as:

$$S_A = \alpha_{\mathrm{stab}} \cdot P \ + \ \alpha_{\mathrm{fid}} \cdot \big(\beta F_t + (1-\beta)F_v\big) \ + \ \alpha_{\mathrm{read}} \cdot \big(\gamma L + (1-\gamma)Ae\big), \tag{6}$$

where $\alpha_{\mathrm{stab}} + \alpha_{\mathrm{fid}} + \alpha_{\mathrm{read}} = 1$ and $\beta, \gamma \in [0,1]$.

3.2 Delivery Scoreboard

Delivery scoreboard measures end-to-end delivery quality beyond static artifacts, mainly focusing on: (i) satisfying requirements and audience constraints, (ii) narrative control, (iii) slide–script complementarity, (iv) temporal pacing, (v) delivery-time guidance, and (vi) rehearsal support.

Requirement satisfaction. R evaluates whether a system's output satisfies the requirement profile $\mathcal{U}$. We decompose R into (i) a fully automatic *timing-compliance* score and (ii) *content-level* requirement checks.

- *Timing compliance.* We compute the timing score from the implied seconds-per-slide s and use a piecewise-linear schedule with a *sweet* range ($12 \le s \le 60$ sec/slide) and a broader *hard* tolerance range ($6 < s < 120$ sec/slide); values outside the hard range receive zero score:

$$R_{\mathrm{time}} = \min\Big\{ \underbrace{1}_{\text{sweet: } 12\le s\le 60},\ \max\Big\{ \underbrace{\min\Big(\frac{s-6}{6}, \frac{120-s}{60}\Big)}_{\text{hard: } 6<s<120}, \underbrace{0}_{\text{outside hard: } s\le 6 \text{ or } s\ge 120} \Big\}\Big\}. \tag{7}$$

- *Richer requirements.* For requirements not captured by simple proxies (e.g., topic coverage, audience fit, and user-specified priorities), we use an LLM-based judging rubric: the judge is prompted with $\mathcal{U}$ and the generated slides/script and returns a scalar compliance score.

Narrative diversity and controllability. We score narrative quality as $N = \eta\, s_{\mathrm{diver}} + (1-\eta)\, s_{\mathrm{ctrl}}$ with $\eta = 0.4$. When multiple narrative-plan candidates are available, s_{diver} measures plan diversity and s_{ctrl} measures narrative controllability. In the single-deck automatic setting (no plan candidates), we set $s_{\mathrm{diver}} = 0$ and estimate s_{ctrl} from two deck-internal cues: (i) the fraction of slides with a non-empty title line, and (ii) the smoothness of topic transitions between consecutive slides, approximated via embedding similarity over slide text and speaker scripts. We aggregate the two cues by averaging, and thus report $N = (1-\eta)\, s_{\mathrm{ctrl}}$.

Complementarity between slide and script. We compute the complementarity score C as the mean of (i) a redundancy "sweet-spot" term (encouraging the script to complement rather than duplicate the slide) and (ii) a lightweight coverage term (encouraging the script to mention key points):

$$C = \frac{1}{2m}\Bigg(\underbrace{\sum_{k=1}^{m}\Big(1 - \frac{\min(|\mathrm{sim}(k) - l|,\ |\mathrm{sim}(k) - u|)}{\max(l, 1-u) + \epsilon}\Big)_+}_{\textit{(i) redundancy sweet-spot}} + \underbrace{\sum_{k=1}^{m}\frac{\#\{\text{words in } f_k \text{ also appearing in } p_k\}}{\#\{\text{words in } f_k\}}}_{\textit{(ii) key-point coverage}} \Bigg), \tag{8}$$

where $\mathrm{sim}(k) \in [0,1]$ is the cosine similarity between slide text of f_k and script p_k. We use a target similarity interval $[l,u] = [0.25, 0.55]$, and $(\cdot)_+ = \max(\cdot, 0)$. Term (i) rewards similarity near the target interval, while (ii) measures the fraction of key content words from f_k that also appear in p_k.

Temporal delivery quality. We estimate per-slide speaking time from script length (150 words/min) and compute Q as the mean of two lightweight proxies:

$$T = \frac{1}{2}\Bigg(\underbrace{\Big(1 - \frac{\mathrm{std}(\{\widehat{d}_k\})}{\mathrm{std}_{\max} + \epsilon}\Big)_+}_{\textit{(i) pacing smoothness}} + \underbrace{\min\Big(1, \frac{\#\text{transition markers}}{4}\Big)}_{\textit{(ii) transitions}} \Bigg). \tag{9}$$

Here $\widehat{d}_k$ is derived from the word count of the script p_k and $\mathrm{std}_{\max}$ is hyperparameter about dataset. Intuitively, (i) rewards balanced per-slide timing, and (ii) rewards explicit transition cues in the script.

Table 1: Main experiment results. Detail metrics are shown in Table 8.

Method	01. AI		02. ML		03. CV		04. NLP		05. Robo		06. Sec		07. SE	
	S_A	S_D	S_A	S_D	S_A	S_D	S_A	S_D	S_A	S_D	S_A	S_D	S_A	S_D
PPTAgent	0.82	0.56	0.80	0.54	0.80	0.54	0.80	0.70	0.83	0.64	0.79	0.64	0.80	0.61
Qwen	0.73	0.61	0.76	0.61	0.75	0.62	0.76	0.59	0.76	0.60	0.73	0.59	0.77	0.60
Coze	0.78	0.50	0.80	0.52	0.81	0.52	0.81	0.49	0.81	0.54	0.74	0.45	0.68	0.41
Gamma	0.84	0.76	0.83	0.76	0.81	0.74	0.81	0.74	0.82	0.76	0.82	0.75	0.80	0.75
Manus	0.82	0.75	0.84	0.75	0.75	0.73	0.82	0.71	0.60	0.58	0.84	0.75	0.84	0.75
NotebookLM	0.75	0.46	0.81	0.49	0.77	0.46	0.80	0.49	0.81	0.50	0.81	0.49	0.75	0.47
DeepSlide	**0.83**	**0.76**	**0.93**	**0.78**	**0.85**	**0.77**	**0.93**	**0.74**	**0.86**	**0.77**	**0.83**	**0.74**	**0.82**	**0.75**

Method	08. Sig		09. Ast		10. HEP		11. CMP		12. Quan		13. Phys		14. Pure	
	S_A	S_D	S_A	S_D	S_A	S_D	S_A	S_D	S_A	S_D	S_A	S_D	S_A	S_D
PPTAgent	0.81	0.59	0.80	0.67	0.80	0.66	0.80	0.64	0.65	0.57	0.80	0.65	0.81	0.72
Qwen	0.75	0.60	0.77	0.63	0.78	0.64	0.77	0.64	0.78	0.65	0.78	0.61	0.73	0.60
Coze	0.80	0.49	0.80	0.54	0.69	0.45	0.73	0.47	0.79	0.50	0.72	0.47	0.75	0.47
Gamma	0.79	0.72	0.83	0.71	0.82	0.76	0.82	0.68	0.81	0.71	0.83	0.76	0.83	0.76
Manus	0.84	0.75	0.84	0.75	0.84	0.76	0.84	0.65	0.84	0.75	0.84	0.76	0.78	0.74
NotebookLM	0.81	0.49	0.81	0.50	0.80	0.49	0.81	0.49	0.81	0.49	0.74	0.47	0.81	0.49
DeepSlide	**0.83**	**0.76**	**0.83**	**0.77**	**0.87**	**0.78**	**0.83**	**0.76**	**0.85**	**0.76**	**0.91**	**0.77**	**0.83**	**0.77**

Method	15. Appl		16. Stats		17. Bio		18. QFin		19. Econ		20. Soc		Avg	
	S_A	S_D	S_A	S_D	S_A	S_D	S_A	S_D	S_A	S_D	S_A	S_D	S_A	S_D
PPTAgent	0.64	0.51	0.80	0.66	0.80	0.70	0.80	0.57	0.63	0.50	0.81	0.50	0.78	0.61
Qwen	0.77	0.63	0.71	0.55	0.78	0.64	0.77	0.61	0.76	0.59	0.77	0.63	0.76	0.61
Coze	0.77	0.46	0.74	0.43	0.76	0.43	0.79	0.48	0.81	0.49	0.76	0.48	0.77	0.48
Gamma	0.70	0.64	0.82	0.74	0.66	0.63	0.84	0.75	0.83	0.75	0.64	0.63	0.80	0.72
Manus	0.84	0.76	0.83	0.75	0.84	0.73	0.84	0.76	0.81	0.73	0.84	0.76	0.82	0.73
NotebookLM	0.81	0.49	0.81	0.49	0.82	0.49	0.82	0.50	0.80	0.49	0.81	0.50	0.80	0.49
DeepSlide	**0.83**	**0.76**	**0.83**	**0.77**	**0.84**	**0.76**	**0.83**	**0.76**	**0.92**	**0.78**	**0.97**	**0.78**	**0.86**	**0.76**

Attention choreography quality and rehearsal readiness. T measures whether delivery-time guidance (e.g., focus/highlight) is synchronized with the speaker's script, while R' measures how close the package is to being "walk-on-stage" ready. We score both metrics in $[0, 1]$ using an LLM-as-judge rubric over slide text and speaker scripts.

Aggregation. To prevent gaming the delivery scoreboard with "flashy but wrong" presentations, we include a portion of Artifact fundamentals:

$$S_D = \sum_{x \in \{R,N,C,T,R'\}} \omega_x \cdot x + \omega_{\text{stab}} \cdot P + \omega_{\text{fid}} \cdot \big(\beta F_t + (1-\beta) F_v\big), \tag{10}$$

3.3 Main Experiment: Domain-wise Evaluation

Fig. 1 visualizes per-domain score profiles over 20 domains. Across domains, DeepSlide consistently achieves strong Delivery scores while remaining competitive on Artifact quality, indicating that optimizing for end-to-end talk delivery does not require sacrificing static slide fundamentals. Detailed metric breakdowns are reported in Table 1 and Table 8.

3.4 Secondary Experiment: Audience-specific Evaluation

Table 9 and Fig 7 report audience-specific results (e.g., engineer, investor, newcomer, researcher, and a mixed group of researcher, engineer and project manager). *DeepSlide* maintains robust delivery performance across audiences, suggesting that its requirement-aware planning and delivery-time support generalize beyond a single audience type. This consistent advantage stems from our audience-oriented narrative-style logical chain recommendation: by first profiling the target listeners and then proposing a time-budgeted storyline that explicitly balances technical depth with

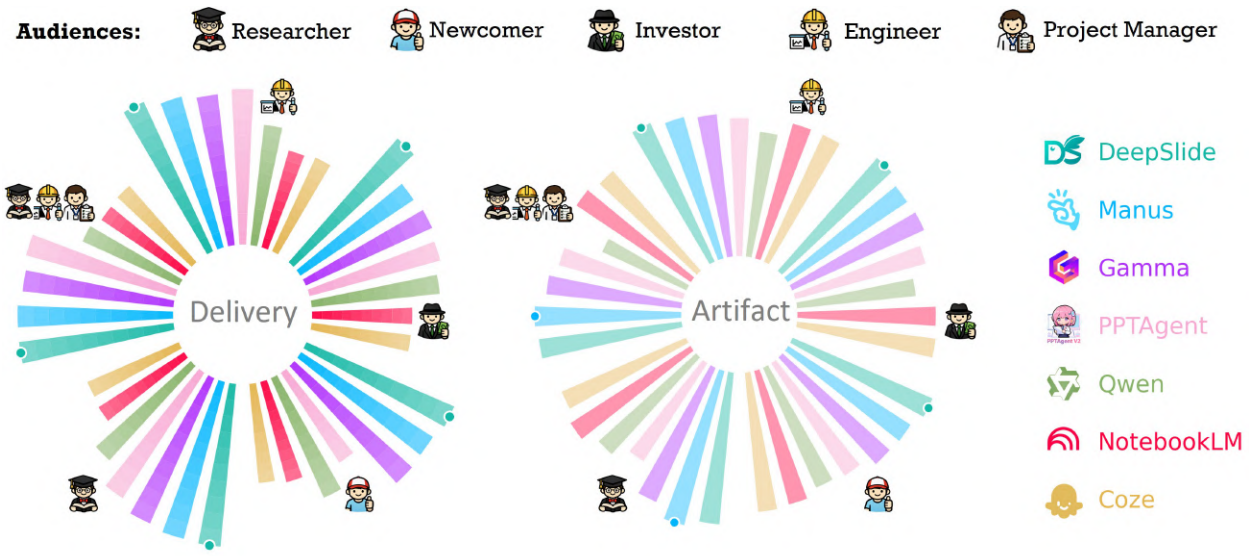


Figure 7: Secondary experiment on audience-specific evaluation.

rhetorical flow, the system ensures that slide sequencing, script emphasis, and visual evidence are all tuned to the cognitive expectations of each audience.

3.5 Case Study and Discussion

Case 1: Is *DeepSlide* merely move content from source to slide? Fig 8 contrasts *DeepSlide* with Manus under varying audience types and duration budgets. DeepSlide exhibits higher stability and more controllable narrative outcomes as requirements change, consistent with its explicit requirement-aware planning and its recommended logic chain that helps users structure a talk under constraints.

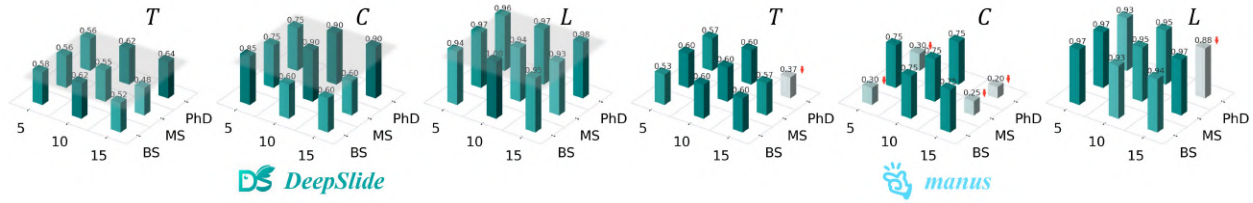


Figure 8: *DeepSlide* vs. Manus, varying audience {BS, MS, PhD} and duration {5, 10, 15} (Case 1).

Case 2: Does DeepSlide reduce user burden and delivery pressure? Table 2 indicates that most presentation agents mainly optimize slide authoring (e.g., outlines and partial source indexing), while leaving delivery and rehearsal effort to users. DeepSlide reduces preparation pressure by making the delivery process explicit and guided. First, requirement elicitation and nonlinear narrative help users lock in a goal-aligned structure earlier, reducing late-stage re-planning. Second, synchronized slide–script generation with explicit complementarity, together with speech support and timing cues, lowers the cognitive load of pacing and deciding what to show versus what to say. Third, preview feedback and audio preview provide a low-cost rehearsal proxy that exposes pacing and rendering issues early, shifting effort from repetitive debugging to small intent-level edits.

Case 3: Which process components drive delivery gains? Table 3 reports an study that removes key pipeline components. Removing the logic chain module yields a substantial drop in Delivery score, while removing the retriever causes a smaller but consistent degradation. suggesting that delivery advantages stem from structured process components (planning, retrieval grounding, and alignment) rather than solely from the underlying model's raw generation capability.

Table 2: Design points across presentation systems.

Design Point	*DeepSlide*	PPTAgent	Qwen	Coze	Gamma	Manus	NotebookLM
Category I: Content planning & structuring (reduce authoring / organization burden)							
Requirement elicitation	✓	✗	✗	✗	✗	✗	✗
Source indexing	✓	✓	○	✗	○	✗	○
Outline	✓	✓	✓	✓	✓	✓	✓
Nonlinear narrative	✓	✗	✗	✗	✗	✗	✗
Category II: Delivery & rehearsal assistance (reduce speaking / rehearsal burden)							
Speech support	✓	✗	✗	✗	✗	✗	✗
Complementarity (slide vs. script)	✓	✗	✗	✗	✗	✗	✗
Timer	✓	✓	✓	✗	✓	✗	✗
Attention strategy	✓	✗	✓	✗	✓	✗	✗
Audio preview	✓	✗	✗	✗	✗	✗	✗
Category III: Interaction & feedback loop (reduce iteration / debugging burden)							
Interactive data visualization	✓	✗	✗	✓	✗	✓	✗
Preview feedback	✓	✗	✗	✗	✗	✗	✗
Audience Question	✓	✗	✗	✗	✗	✗	✗

✓: supported ✗: not supported ○: unknown

Table 3: Impact of removing key components on the Delivery Scoreboard S_D (Case 3).

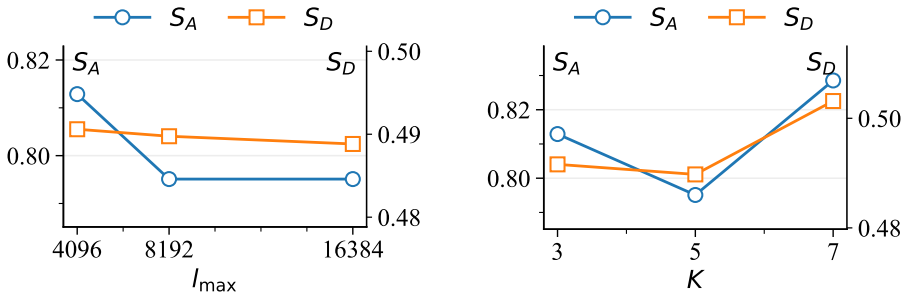
Method	S_D	P	C	T	N
DeepSlide	**0.68**	**0.75**	**0.86**	**0.61**	**0.55**
w/o BM25 content tree retriever	0.67 (-0.01)	0.74 (-0.01)	0.85 (-0.01)	0.59 (-0.02)	0.54 (-0.01)
w/o logical chain	0.44 (-0.24)	0.12 (-0.63)	0.10 (-0.76)	0.36 (-0.25)	0.53 (-0.02)
w/o logical chain recommender	0.68 (-0.00)	0.71 (-0.04)	0.85 (-0.01)	0.60 (-0.01)	0.53 (-0.02)

3.5.1 Ablation Study and Discussion

We will extend the ablation suite with additional variants and controlled settings in future revisions.

Varying truncation length. We vary truncation length $l_{\max} \in \{4096, 8192, 16384\}$. S_A peaks at 4096 (0.813) and then stabilizes at ≈ 0.795 as $l_{\max}$ increases, suggesting tighter contexts reduce retrieval noise. S_D is nearly unchanged ($0.491 \rightarrow 0.489$), indicating negligible impact on delivery quality.

Varying retrieval depth. We vary retrieval depth $K \in \{3, 5, 7\}$. Ablation performance on CV papers is non-monotonic: the default $K = 5$ underperforms both $K = 3, 7$. With $K = 3$, the model maintains high artifact quality by focusing on the most relevant nodes ($S_A = 0.813$), whereas $K = 5$ injects noise with limited gain ($S_A = 0.795$). Increasing to $K = 7$ yields the best overall scores ($S_A = 0.829, S_D = 0.503$) and improves visual recall ($F_v : 0.0 \rightarrow 0.125$). This suggests that in CV domain, complementary evidence (e.g., ablations and qualitative



(a) Varying truncation length. (b) Varying retrieval depth.

Figure 9: Varying retrieval depth.

figures) is often dispersed across sections; broader retrieval better recovers these dependencies and strengthens grounding.

4 Conclusion

We presented *DeepSlide*, a delivery-oriented presentation preparation system that unifies planning, generation, and rehearsal in one pipeline. It integrates time-budgeted logical chains, content-tree grounded retrieval, Markov-style sequential rendering with style inheritance, and sandbox validation with minimal repair to improve end-to-end deliverability. We also introduced a dual-scoreboard protocol that separates artifact quality from delivery quality, enabling targeted diagnosis of narrative, pacing, and slide–script coordination failures. Across both the domain study and the audience-profile study, *DeepSlide* matches strong baselines on artifact quality while delivering consistent, substantial gains on delivery metrics, yielding more reliable narrative flow, pacing precision, and attention guidance across diverse settings.

References

[1] Alan Kelly. *How Scientists Communicate: Dispatches from the Frontiers of KnowledgeDispatches from the Frontiers of Knowledge*. 09 2020. ISBN 9780190936600. doi: 10.1093/oso/9780190936600.001.0001.

[2] Michael Alley. The craft of scientific presentations: Critical steps to succeed and critical errors to avoid. *Physics Today*, 57, 07 2004. doi: 10.1063/1.1784305.

[3] Jiaxin Ge, Zora Zhiruo Wang, Xuhui Zhou, Yi-Hao Peng, Sanjay Subramanian, Qinyue Tan, Maarten Sap, Alane Suhr, Daniel Fried, Graham Neubig, et al. Autopresent: Designing structured visuals from scratch. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 2902–2911, 2025.

[4] Himanshu Maheshwari, Sambaran Bandyopadhyay, Aparna Garimella, and Anandhavelu Natarajan. Presentations are not always linear! gnn meets llm for document-to-presentation transformation with attribution. *arXiv preprint arXiv:2405.13095*, 2024.

[5] Ishani Mondal, Shwetha S, Anandhavelu Natarajan, Aparna Garimella, Sambaran Bandyopadhyay, and Jordan Boyd-Graber. Presentations by the humans and for the humans: Harnessing LLMs for generating persona-aware slides from documents. In Yvette Graham and Matthew Purver, editors, *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2664–2684, St. Julian's, Malta, March 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.eacl-long.163. URL https://aclanthology.org/2024.eacl-long.163/.

[6] langchain-ai/langchain: The platform for reliable agents. https://github.com/langchain-ai/langchain, 2026. Accessed: 2026-02-26.

[7] Significant-gravitas/autogpt: Autogpt is the vision of accessible ai for everyone, to use and to build on. our mission is to provide the tools, so that you can focus on what matters. https://github.com/Significant-Gravitas/AutoGPT, 2026. Accessed: 2026-02-26.

[8] microsoft/autogen: A programming framework for agentic ai. https://github.com/microsoft/autogen, 2026. Accessed: 2026-02-26.

[9] Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. Camel: Communicative agents for "mind" exploration of large language model society. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[10] Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. MetaGPT: Meta programming for a multi-agent collaborative framework. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=VtmBAGCN7o.

[11] Manus. https://manus.im/app, 2026. Accessed: 2026-02-26.

[12] Gamma. https://gamma.app/, 2026. Accessed: 2026-02-26.

[13] Google notebooklm. https://notebooklm.google/, 2026. Accessed: 2026-02-26.

[14] Qwen. https://qwen.ai/home, 2026. Accessed: 2026-02-26.

[15] Coze: Next-gen ai app developing platform. https://www.coze.com/, 2026. Accessed: 2026-02-26.

[16] Hao Zheng, Xinyan Guan, Hao Kong, Wenkai Zhang, Jia Zheng, Weixiang Zhou, Hongyu Lin, Yaojie Lu, Xianpei Han, and Le Sun. PPTAgent: Generating and evaluating presentations beyond text-to-slides. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng, editors, *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 14402–14418, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.728. URL https://aclanthology.org/2025.emnlp-main.728/.

[17] Richard E. Mayer. *Coherence Principle*, page 113–133. Cambridge University Press, 2001.

[18] Tushar Aggarwal and Aarohi Bhand. Pass: Presentation automation for slide generation and speech. *arXiv preprint arXiv:2501.06497*, 2025.

[19] Xiaojie Xu, Xinli Xu, Sirui Chen, Haoyu Chen, Fan Zhang, and Ying-Cong Chen. Pregenie: An agentic framework for high-quality visual presentation generation. *arXiv preprint arXiv:2505.21660*, 2025.

[20] Isabel Cachola, Silviu Cucerzan, Allen Herring, Vuksan Mijovic, Erik Oveson, and Sujay Kumar Jauhar. Knowledge-centric templatic views of documents. *arXiv preprint arXiv:2401.06945*, 2024.

[21] Yunqiao Yang, Wenbo Li, Houxing Ren, Zimu Lu, Ke Wang, Zhiyuan Huang, Zhuofan Zong, Mingjie Zhan, and Hongsheng Li. Slidesgen-bench: Evaluating slides generation via computational and quantitative metrics. *arXiv preprint arXiv:2601.09487*, 2026.

[22] Michael Ofengenden, Yunze Man, Ziqi Pang, and Yu-Xiong Wang. Pptarena: A benchmark for agentic powerpoint editing. *arXiv preprint arXiv:2512.03042*, 2025.

[23] Zheng Huang, Xukai Liu, Tianyu Hu, Kai Zhang, and Ye Liu. Pptbench: Towards holistic evaluation of large language models for powerpoint layout and design understanding. *arXiv preprint arXiv:2512.02624*, 2025.

[24] Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

[25] Siyi Zhou, Yiquan Zhou, Yi He, Xun Zhou, Jinchao Wang, Wei Deng, and Jingchen Shu. Indextts2: A breakthrough in emotionally expressive and duration-controlled auto-regressive zero-shot text-to-speech. *arXiv preprint arXiv:2506.21619*, 2025.

[26] Dayuanjiang/next-ai-draw-io: A next.js web application that integrates ai capabilities with draw.io diagrams. this app allows you to create, modify, and enhance diagrams through natural language commands and ai-assisted visualization. https://github.com/DayuanJiang/next-ai-draw-io, 2026. Accessed: 2026-02-27.

[27] Deqing Li, Honghui Mei, Yi Shen, Shuang Su, Wenli Zhang, Junting Wang, Ming Zu, and Wei Chen. Echarts: A declarative framework for rapid construction of web-based visualization. *Visual Informatics*, 2(2):136–146, 2018. ISSN 2468-502X. doi: https://doi.org/10.1016/j.visinf.2018.04.011. URL https://www.sciencedirect.com/science/article/pii/S2468502X18300068.

[28] Gemini image – nano banana — google deepmind. https://deepmind.google/models/gemini-image/, 2026. Accessed: 2026-02-27.

[29] Tsu-Jui Fu, William Yang Wang, Daniel McDuff, and Yale Song. Doc2ppt: Automatic presentation slides generation from scientific documents. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 634–642, 2022.

[30] Keshav Kumar and Ravindranath Chowdary. Slidespawn: An automatic slides generation system for research publications. *arXiv preprint arXiv:2411.17719*, 2024.

[31] Yuheng Yang, Wenjia Jiang, Yang Wang, Yiwei Wang, and Chi Zhang. Auto-slides: An interactive multi-agent system for creating and customizing research presentations. *arXiv preprint arXiv:2509.11062*, 2025.

[32] Jingwei Shi, Zeyu Zhang, Biao Wu, Yanjie Liang, Meng Fang, Ling Chen, and Yang Zhao. PresentAgent: Multimodal agent for presentation video generation. In Ivan Habernal, Peter Schulam, and Jörg Tiedemann, editors, *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 760–773, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-334-0. doi: 10.18653/v1/2025.emnlp-demos.58. URL https://aclanthology.org/2025.emnlp-demos.58/.

[33] Alexander Meier, Mahei Manhai Li, and Roman Rietsche. Developing a hybrid vector-graph retrieval system for entity-preserving and inspiring storyline creation of presentation slides, 2025.

[34] X. He, Y. Zhang, L. Wang, and Q. Liu. A survey on large language models for narrative visualization. *arXiv preprint arXiv:2405.12345*, 2024.

[35] Microsoft 365 copilot | ai productivity tools for work. https://www.microsoft.com/en-us/microsoft-365-copilot, 2026. Accessed: 2026-02-26.

[36] Google gemini. https://gemini.google.com/app, 2026. Accessed: 2026-02-26.

[37] Best ai presentation maker for professional decks | beautiful.ai - generate high-quality slides with the artificial intelligence powered presentation tool available. https://www.beautiful.ai/, 2026. Accessed: 2026-02-26.

[38] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

[39] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

[40] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*, 2019.

[41] Yiduo Guo, Zekai Zhang, Yaobo Liang, Dongyan Zhao, and Nan Duan. Pptc benchmark: Evaluating large language models for powerpoint task completion. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 8682–8701, 2024.

[42] Ananth Muppidi, Tarak Das, Sambaran Bandyopadhyay, Tripti Shukla, et al. Taming llms with negative samples: A reference-free framework to evaluate presentation content with actionable feedback. *arXiv preprint arXiv:2505.18240*, 2025.

Table of Contents for Appendices

Appendix A System Details

DeepSlide's system implementation and LLM usage are detailed in Table 4 and Table 6. LLMs were also used for polishing and copy-editing parts of the manuscript; the authors are fully responsible for all content.

Table 4: Stage-wise agents and their responsibilities.

Agent	Responsibility
Stage 1: Requirement elicitation and narrative proposal	
requirements collector	elicit audience, duration, and constraints into a requirement profile.
narrative template selector	select a narrative template consistent with the requirement profile.
logical chain generator	produce a time-budgeted logical chain (or chain graph) for the talk.
Stage 2: Logical chain editing and evidence-grounded generation	
logical edge generator	add sparse cross-node links for long-range coherence and bridging.
semantic matcher	align paper sections/content tree nodes to logical nodes for retrieval.
slide generator	retrieve evidence and synthesize slides with scripts under time budgets.
compiler debugger	iteratively repair compilation errors until the deck compiles successfully.
Stage 3: Interactive slide refinement and attention-oriented augmentation	
style agent	choose a deck-level style prior to reduce cross-slide drift.
visual intent agent	specify visual goals and negative constraints (what should *not* be drawn).
diagram spec agent	extract diagram semantics into a structured specification.
render plan agent	produce an executable plan (layout/style/effects/assets) for rendering.
drawio xml generator	emit editable drawio xml from the diagram specification.
render reviewer	execute the deck in a browser sandbox and propose minimal fix plan.
coherence analyzer	inject lightweight cross-slide references when needed.
vlm design expert	resolve image–code inconsistencies via minimal frame/layout edits.
editor planner	translate user edits into a bounded action plan.
content editor	apply localized edits on targeted frames/titles/scripts.
Stage 4: Rehearsal and dual-scoreboard evaluation	
rehearsal coach	provide actionable delivery tips from rehearsal metrics and context.
audience reviewer	generate likely audience questions to stress-test coverage and clarity.

Appendix B Evaluation

B.1 Baselines

Baselines. We compare *DeepSlide* against six representative systems: PPTAgent [16], Manus [11], Coze [15], Gamma [12], NotebookLM [13], and Qwen [14]. PPTAgent is open-source, while the others are closed-source commercial products. Detailed baseline descriptions and usage settings are reported in Table 5. For Manus, Coze, and NotebookLM, which can generate slide content directly as images (e.g., via an image model such as NanoBanana [28]), we enable this image-based mode as the baseline setting. Image-as-slide generation has become increasingly adopted, yet it still suffers from limited editability and imprecise element-level control.

Table 5: Baseline systems used for comparison.

System	Open Source	License	Version&Date	URL
Manus	✗	NA	Manus 1.6 Max Pro	`https://manus.im/app`
Gamma	✗	NA	Gamma Pro	`https://gamma.app/`
PPTAgent	✓	MIT license	PPTAgent-V2	`https://github.com/icip-cas/PPTAgent`
Qwen	✗	NA	Qwen3-Max	`https://chat.qwen.ai/`
NoteBookLM	✗	NA	NoteBookLM Pro	`https://notebooklm.google.com/`
Coze	✗	NA	Jan 13–16, 2026	`https://www.coze.com/`

API usage. *DeepSlide* supports per-agent API configurations and also provides default grouped presets. For high-usage agents with modest intelligence requirements, we assign low-cost yet sufficiently capable models (e.g., deepseek-

chat, qwen3-vl-flash). For agents that most strongly affect the Artifact/Delivery scoreboards, we use more capable models (e.g., gpt-5-mini, gemini-3-pro-preview). Table 6 summarizes the configurations of *DeepSlide* and PPTAgent.

Table 6: API usage of *DeepSlide* and PPTAgent [16].

Agents&System	**API**
DeepSlide	
Stage 1: Requirement elicitation and narrative proposal	
requirements collector	deepseek-chat
narrative template selector	deepseek-chat
logical chain generator	deepseek-chat
Stage 2: Logical chain editing and evidence-grounded generation	
logical edge generator	deepseek-chat
semantic matcher	deepseek-chat
slide generator	gpt-5-mini
compiler debugger	deepseek-chat
Stage 3: Interactive slide refinement and attention-oriented augmentation	
style agent	deepseek-chat
visual intent agent	deepseek-chat
diagram spec agent	deepseek-chat
render plan agent	deepseek-chat
drawio xml generator	deepseek-chat
render reviewer	gemini-3-pro-preview
coherence analyzer	deepseek-chat
vlm design expert	qwen3-vl-flash
editor planner	deepseek-chat
content editor	deepseek-chat
Stage 4: Rehearsal and dual-leaderboard evaluation	
rehearsal coach	deepseek-chat
audience reviewer	deepseek-chat
PPTAgent	
research_agent	gpt-5-mini
design_agent	gpt-5-mini
long_context_model	gpt-5-mini
vision_model	gpt-5-mini
pptagent	gpt-5-mini

B.2 Dataset and Prompts

Domains. Our main benchmark spans 20 domains to stress-test presentation agents under diverse evidence structures and writing conventions, ranging from visually dense experimental papers (e.g., CV and Sig) to theorem-heavy manuscripts (e.g., Pure and Appl) and narrative-driven social-science writing (Soc). Table 7 lists the domain taxonomy used throughout the paper.

Table 7: The 20 domains covered in the benchmark and the dominant evidence patterns that shape slide design and delivery strategy.

Abbrev.	Domain	Typical evidence patterns in papers
COMPUTER SCIENCE & ENGINEERING		
AI	Artificial Intelligence	Studies intelligent agents and reasoning for real-world tasks.
ML	Machine Learning	Develops learning algorithms and theory from data at scale.
CV	Computer Vision	Understands images and videos.
NLP	Natural Language Processing	Generates human language for understanding and interaction.
Robo	Robotics and Control Systems	Autonomous robots with sensing, control, and planning.
Sec	Cryptography and Security	Ensures confidentiality, and robustness against adversaries.
SE	Software Engineering and Systems	Builds scalable software with performance and reliability.
Sig	Signal and Image Processing	Analyzes signals using transforms and statistical methods.
PHYSICS		
Ast	Astrophysics and Cosmology	Explores universe using observations, simulations, and theory.
HEP	High-Energy Physics	Probes fundamental particles with accelerators/detectors.
CMP	Condensed Matter Physics	Studies matter phases and properties in quantum.
Quan	Quantum Physics	Investigates quantum information, circuits, and complexity.
Phys	General Physics	Broad physical principles across theory and modeling.
MATHEMATICS & STATISTICS		
Pure	Pure Mathematics	Proves abstract structures through definitions and theorems.
Appl	Applied Mathematics and Analysis	Models real phenomena with analysis, numerics, etc.
Stats	Statistics and Probability	Inference and uncertainty quantification for data/experiments.
ECONOMICS, FINANCE & SOCIETY		
Bio	Quantitative Biology	Quantifies biological systems with experiments, models, etc.
QFin	Quantitative Finance	Models markets with stochastic processes and empirical tests.
Econ	Economics and Econometrics	Studies markets using theory and econometric models.
Soc	Social Information Networks	Analyzes social and networked behavior with data.

Prompt usage. Each benchmark instance pairs a source paper $\mathcal{T}$ with a requirement profile $\mathcal{U}$. In our experiments, prompts are grouped by how $\mathcal{U}$ is instantiated: (i) the main experiment uses the *Researcher* profile as the default prompt block; (ii) the secondary experiment swaps only the **Audience** field to one of five cases (Engineer, Investor, Newcomer, Researcher, Hybrid) while keeping the remaining fields consistent; (iii) the case study Case 2/Case 3 and ablations also use the *Researcher* profile, whereas case study Case 1 varies audience background levels (BS/MS/PhD) and time budgets. We highlight the four user-facing fields in **color**: **Audience**, **Duration**, **Focus**, and **Style**.

Prompt Block M: Researcher Profile (main experiment; Q2/Q3; ablations; also used as the "Researcher").
Audience: researchers in the field
Duration: {D} minutes
Focus: novelty; assumptions; comparisons; ablations; theoretical or empirical claims
Style: rigorous, citation-aware, emphasize evidence and reproducibility

Instruction: Clearly separate contributions from background, highlight what is new, and justify design choices with ablations or analysis. Keep slides concise, avoid redundancy between on-slide text and script, and make delivery transitions explicit.

Prompt Block R1: Audience = Engineer (secondary experiment).
Audience: senior engineers and applied ML practitioners
Duration: {D} minutes
Focus: system design; implementation details; failure modes; deployment trade-offs
Style: technical, structured, actionable; prefer diagrams and concrete examples

Instruction: Emphasize how the method works end-to-end, what to implement first, and what to watch out for. Use clear module boundaries and discuss cost/latency/robustness when applicable.

Prompt Block R2: Audience = Investor (secondary experiment).
Audience: investors and product decision makers
Duration: {D} minutes
Focus: value proposition; differentiation; risks; expected impact and roadmap
Style: high-level, persuasive but evidence-backed; minimize technical notation

Instruction: Start from the problem and why it matters, then present the key idea and evidence of effectiveness. Translate metrics into outcomes, and explicitly state limitations and go-to-market risks.

Prompt Block R3: Audience = Newcomer (secondary experiment).
Audience: newcomers with basic ML background
Duration: {D} minutes
Focus: intuition; definitions; simplified workflow; one main takeaway per section
Style: gentle pacing, minimal jargon, use analogies and progressive disclosure

Instruction: Explain prerequisites briefly, define key terms before using them, and prioritize conceptual clarity over exhaustive details. Use fewer equations and more visual/step-based explanations.

Prompt Block R4: Audience = Researcher+Engineer+PM (Hybrid) (secondary experiment).
Audience: mixed audience (research, engineering, product)
Duration: {D} minutes
Focus: core idea; engineering feasibility; product implications; evaluation evidence
Style: balanced depth, executive clarity, and technical grounding

Instruction: Present the narrative in three layers: what it is (intuition), why it works (key mechanism), and how to use it (integration and constraints). Use one slide to summarize practical takeaways and risks.

Prompt Block C: Case Study Q1 (audience background level and time budget sweeps).
Audience: {BS-level / MS-level / PhD-level}
Duration: $\{D_1, D_2, \dots\}$ minutes
Focus: controllable narrative under constraints; robustness to requirement changes
Style: requirement-aware, structured, avoid over-specialization beyond the target level

Instruction: Adapt depth, terminology, and the amount of background to the specified audience level. Under the given duration, prioritize the most important contributions and keep transitions explicit.

B.3 Experimental Results

Experimental Setup. We evaluate presentation agents under the proposed dual-scoreboard protocol (Section 3). Unless otherwise noted, each system is run $\tau = 5$ times per instance and we report mean scores.

Table 8: Supplementary detailed metrics.

Method	01. AI								02. ML							
	P	F_t	F_v	L	R	C	T	N	P	F_t	F_v	L	R	C	T	N
DeepSlide	1.00	0.95	1.00	0.99	0.74	0.85	0.57	0.54	1.00	0.95	0.67	1.00	0.74	0.85	0.61	0.54
PPTAgent	1.00	0.90	0.33	0.90	0.55	0.27	0.27	0.41	1.00	0.93	0.00	0.95	0.71	0.24	0.55	0.55
Qwen	1.00	0.72	0.00	0.83	0.40	0.44	0.51	0.52	1.00	0.84	0.00	0.84	0.50	0.34	0.48	0.52
Coze	1.00	0.91	0.00	0.89	0.72	0.18	0.57	0.55	1.00	0.92	0.00	0.98	0.74	0.30	0.59	0.55
Gamma	1.00	0.95	0.00	0.98	0.74	0.85	0.60	0.55	1.00	0.95	0.05	0.94	0.74	0.85	0.61	0.55
Manus	1.00	0.95	0.25	0.94	0.74	0.84	0.60	0.55	1.00	0.95	0.00	1.00	0.74	0.85	0.63	0.54

NotebookLM	1.00	0.75	0.00	1.00	0.64	0.12	0.58	0.55	1.00	0.94	0.00	0.99	0.73	0.12	0.61	0.56

Method	**03. CV**								**04. NLP**							
	P	F_t	F_v	L	R	C	T	N	P	F_t	F_v	L	R	C	T	N
DeepSlide	1.00	0.95	0.12	1.00	0.74	0.85	0.60	0.54	1.00	0.90	0.75	0.99	0.71	0.70	0.57	0.56
PPTAgent	1.00	0.92	0.26	0.95	0.71	0.24	0.57	0.54	1.00	0.93	0.03	0.85	0.73	0.67	0.60	0.55
Qwen	1.00	0.80	0.00	0.84	0.49	0.40	0.52	0.52	1.00	0.82	0.00	0.84	0.35	0.36	0.48	0.51
Coze	1.00	0.93	0.03	0.99	0.73	0.31	0.59	0.55	1.00	0.93	0.00	0.99	0.73	0.16	0.59	0.54
Gamma	1.00	0.91	0.00	0.90	0.72	0.76	0.60	0.55	1.00	0.94	0.00	0.89	0.73	0.73	0.59	0.55
Manus	1.00	0.90	0.00	0.69	0.71	0.75	0.60	0.55	1.00	0.90	0.00	0.98	0.71	0.70	0.60	0.54
NotebookLM	1.00	0.80	0.03	0.99	0.64	0.14	0.55	0.56	1.00	0.92	0.00	0.99	0.72	0.12	0.59	0.55

Method	**05. Robo**								**06. Sec**							
	P	F_t	F_v	L	R	C	T	N	P	F_t	F_v	L	R	C	T	N
DeepSlide	1.00	0.95	0.19	1.00	0.74	0.85	0.57	0.55	1.00	0.95	–	1.00	0.74	0.85	0.44	0.55
PPTAgent	1.00	0.95	0.04	0.99	0.72	0.46	0.60	0.56	1.00	0.95	0.05	0.85	0.73	0.46	0.60	0.55
Qwen	1.00	0.84	0.00	0.84	0.40	0.32	0.50	0.51	1.00	0.72	0.00	0.85	0.39	0.36	0.48	0.51
Coze	1.00	0.95	0.00	0.97	0.74	0.40	0.60	0.55	1.00	0.73	0.00	0.99	0.65	0.12	0.55	0.55
Gamma	1.00	0.95	0.02	0.91	0.73	0.86	0.60	0.56	1.00	0.94	0.00	0.90	0.73	0.82	0.60	0.56
Manus	1.00	0.18	0.00	1.00	1.00	0.22	0.33	0.55	1.00	0.95	–	1.00	0.74	0.85	0.60	0.54
NotebookLM	1.00	0.92	0.00	1.00	0.73	0.16	0.59	0.56	1.00	0.92	0.00	1.00	0.73	0.10	0.59	0.56

Method	**07. SE**								**08. Sig**							
	P	F_t	F_v	L	R	C	T	N	P	F_t	F_v	L	R	C	T	N
DeepSlide	1.00	0.90	–	1.00	0.71	0.80	0.57	0.56	1.00	0.95	–	1.00	0.75	0.85	0.59	0.54
PPTAgent	1.00	0.95	0.08	0.90	0.73	0.43	0.60	0.54	1.00	0.93	0.24	0.95	0.72	0.32	0.60	0.56
Qwen	1.00	0.86	0.00	0.84	0.37	0.38	0.49	0.51	1.00	0.79	0.00	0.84	0.35	0.44	0.49	0.52
Coze	1.00	0.57	0.00	0.95	0.55	0.08	0.52	0.55	1.00	0.89	0.00	0.99	0.72	0.16	0.57	0.56
Gamma	1.00	0.94	0.00	0.84	0.73	0.84	0.60	0.55	1.00	0.92	0.12	0.81	0.73	0.62	0.59	0.56
Manus	1.00	0.95	–	1.00	0.74	0.85	0.60	0.55	1.00	0.95	0.00	1.00	0.74	0.85	0.60	0.54
NotebookLM	1.00	0.74	0.00	0.99	0.72	0.10	0.58	0.56	1.00	0.92	0.00	1.00	0.74	0.12	0.59	0.55

Method	**09. Ast**								**10. HEP**							
	P	F_t	F_v	L	R	C	T	N	P	F_t	F_v	L	R	C	T	N
DeepSlide	1.00	0.95	–	1.00	0.77	0.90	0.58	0.55	1.00	0.95	1.00	1.00	0.77	0.90	0.59	0.55
PPTAgent	1.00	0.94	0.17	0.85	0.73	0.49	0.59	0.55	1.00	0.94	0.18	0.89	0.74	0.57	0.60	0.56
Qwen	1.00	0.86	0.00	0.84	0.49	0.40	0.51	0.53	1.00	0.88	0.00	0.83	0.47	0.53	0.49	0.52
Coze	1.00	0.94	0.00	0.95	0.75	0.41	0.61	0.54	1.00	0.55	0.00	1.00	0.75	0.08	0.59	0.56
Gamma	1.00	0.93	0.00	0.95	0.73	0.55	0.58	0.56	1.00	0.95	0.11	0.91	0.75	0.87	0.61	0.55
Manus	1.00	0.95	–	1.00	0.74	0.85	0.63	0.51	1.00	0.95	0.00	1.00	0.77	0.90	0.62	0.56
NotebookLM	1.00	0.93	0.00	1.00	0.74	0.14	0.60	0.56	1.00	0.91	0.00	1.00	0.75	0.12	0.59	0.56

Method	**11. CMP**								**12. Quan**							
	P	F_t	F_v	L	R	C	T	N	P	F_t	F_v	L	R	C	T	N
DeepSlide	1.00	0.95	–	1.00	0.74	0.85	0.60	0.55	1.00	0.95	0.14	1.00	0.71	0.90	0.57	0.54
PPTAgent	1.00	0.95	0.00	0.86	0.74	0.45	0.60	0.56	0.80	0.73	0.06	0.75	0.59	0.44	0.47	0.45
Qwen	1.00	0.86	0.00	0.84	0.45	0.52	0.47	0.53	1.00	0.88	0.00	0.84	0.49	0.54	0.52	0.52
Coze	1.00	0.74	0.00	0.92	0.64	0.14	0.56	0.56	1.00	0.92	0.00	0.93	0.74	0.12	0.58	0.55
Gamma	1.00	0.94	0.00	0.92	0.72	0.37	0.58	0.55	1.00	0.92	0.00	0.90	0.62	0.70	0.57	0.55
Manus	1.00	0.95	0.00	1.00	0.74	0.20	0.60	0.52	1.00	0.95	–	1.00	0.74	0.85	0.60	0.55
NotebookLM	1.00	0.94	0.00	1.00	0.74	0.12	0.60	0.55	1.00	0.92	0.00	1.00	0.75	0.12	0.60	0.57

Method	13. Phys								14. Pure							
	P	F_t	F_v	L	R	C	T	N	P	F_t	F_v	L	R	C	T	N
DeepSlide	1.00	0.95	1.00	1.00	0.74	0.85	0.57	0.54	1.00	0.95	–	1.00	0.74	0.85	0.60	0.53
PPTAgent	1.00	0.92	0.33	0.90	0.74	0.56	0.60	0.55	1.00	0.93	–	0.89	0.71	0.62	0.61	0.54
Qwen	1.00	0.88	0.00	0.84	0.43	0.36	0.45	0.52	1.00	0.74	–	0.84	0.37	0.44	0.45	0.52
Coze	1.00	0.71	0.17	0.93	0.64	0.20	0.53	0.53	1.00	0.77	–	0.96	0.64	0.22	0.55	0.55
Gamma	1.00	0.94	0.00	0.94	0.75	0.84	0.61	0.56	1.00	0.95	–	0.96	0.75	0.87	0.61	0.54
Manus	1.00	0.95	0.00	1.00	0.77	0.90	0.62	0.54	1.00	0.80	–	0.94	0.74	0.85	0.63	0.56
NotebookLM	1.00	0.70	0.08	1.00	0.76	0.10	0.58	0.57	1.00	0.93	–	1.00	0.74	0.10	0.60	0.56

Method	15. Appl								16. Stats							
	P	F_t	F_v	L	R	C	T	N	P	F_t	F_v	L	R	C	T	N
DeepSlide	1.00	0.95	–	1.00	0.74	0.85	0.57	0.54	1.00	0.95	–	1.00	0.77	0.90	0.58	0.51
PPTAgent	0.80	0.73	–	0.75	0.59	0.40	0.49	0.45	1.00	0.92	–	0.90	0.74	0.56	0.60	0.57
Qwen	1.00	0.86	–	0.83	0.41	0.50	0.50	0.52	1.00	0.67	–	0.84	0.28	0.28	0.43	0.51
Coze	1.00	0.82	–	0.99	0.69	0.08	0.56	0.55	1.00	0.72	–	0.99	0.56	0.12	0.50	0.55
Gamma	1.00	0.57	–	0.87	0.66	0.45	0.56	0.54	1.00	0.94	–	0.90	0.75	0.73	0.60	0.55
Manus	1.00	0.95	–	1.00	0.77	0.90	0.62	0.52	1.00	0.95	–	0.98	0.77	0.90	0.62	0.49
NotebookLM	1.00	0.92	–	1.00	0.75	0.08	0.61	0.56	1.00	0.92	–	1.00	0.72	0.10	0.62	0.55

Method	17. Bio								18. QFin							
	P	F_t	F_v	L	R	C	T	N	P	F_t	F_v	L	R	C	T	N
DeepSlide	1.00	0.95	0.00	1.00	0.74	0.85	0.56	0.55	1.00	0.95	–	1.00	0.74	0.85	0.58	0.55
PPTAgent	1.00	0.94	0.00	0.85	0.73	0.66	0.60	0.54	1.00	0.95	0.58	0.90	0.74	0.20	0.59	0.54
Qwen	1.00	0.88	0.00	0.84	0.46	0.46	0.55	0.52	1.00	0.86	0.00	0.83	0.42	0.36	0.51	0.52
Coze	1.00	0.78	0.00	0.99	0.55	0.08	0.51	0.55	1.00	0.90	0.00	0.95	0.70	0.12	0.55	0.53
Gamma	1.00	0.40	0.00	0.95	0.79	0.34	0.55	0.55	1.00	0.95	0.00	0.97	0.76	0.77	0.60	0.54
Manus	1.00	0.95	0.00	1.00	0.74	0.70	0.60	0.57	1.00	0.95	–	1.00	0.77	0.90	0.62	0.56
NotebookLM	1.00	0.95	0.00	1.00	0.74	0.10	0.59	0.56	1.00	0.95	0.00	1.00	0.75	0.12	0.61	0.56

Method	19. Econ								20. Soc							
	P	F_t	F_v	L	R	C	T	N	P	F_t	F_v	L	R	C	T	N
DeepSlide	1.00	0.95	0.60	1.00	0.74	0.85	0.61	0.56	1.00	0.95	1.00	1.00	0.74	0.85	0.61	0.55
PPTAgent	0.80	0.70	0.40	0.74	0.59	0.41	0.45	0.44	1.00	0.93	0.07	1.00	0.69	0.16	0.59	0.56
Qwen	1.00	0.84	0.00	0.84	0.39	0.28	0.48	0.52	1.00	0.86	0.00	0.84	0.47	0.42	0.50	0.52
Coze	1.00	0.92	0.00	1.00	0.72	0.12	0.58	0.54	1.00	0.80	0.01	0.98	0.65	0.30	0.52	0.55
Gamma	1.00	0.93	0.00	0.97	0.73	0.83	0.61	0.55	1.00	0.42	0.04	0.84	0.79	0.33	0.54	0.55
Manus	1.00	0.85	0.00	0.98	0.74	0.70	0.60	0.56	1.00	0.95	0.00	1.00	0.77	0.90	0.62	0.56
NotebookLM	1.00	0.90	0.00	0.99	0.75	0.12	0.61	0.55	1.00	0.95	0.00	0.99	0.74	0.18	0.60	0.56

Overall Average

Method	Average (20 Domains)							
	P	F_t	F_v	L	R	C	T	N
DeepSlide	1.00	0.94	0.59	1.00	0.74	0.85	0.58	0.54
PPTAgent	0.97	0.90	0.17	0.88	0.70	0.43	0.56	0.53
Qwen	1.00	0.82	0.00	0.84	0.42	0.41	0.49	0.52
Coze	1.00	0.82	0.01	0.97	0.68	0.18	0.56	0.55
Gamma	1.00	0.87	0.02	0.91	0.73	0.70	0.59	0.55
Manus	1.00	0.89	0.02	0.98	0.76	0.77	0.60	0.54
NotebookLM	1.00	0.89	0.01	1.00	0.73	0.12	0.59	0.56

Table 9: Audience-specific dual-scoreboard results (Artifact v.s. Delivery). DeepSlide rows are highlighted.

Method	Artifact					Delivery					Artifact					Delivery				
	S_A	P	F_t	F_v	L	S_D	R	C	T	N	S_A	P	F_t	F_v	L	S_D	R	C	T	N
Engineer											*Investor*									
DeepSlide	**0.87**	1.00	0.95	1.00	1.00	**0.79**	0.71	0.90	0.57	0.56	**0.86**	1.00	0.90	1.00	1.00	**0.79**	0.74	0.70	0.79	0.56
PPTAgent	0.79	1.00	0.95	1.00	1.00	0.74	0.74	0.85	0.63	0.55	0.77	1.00	0.90	1.00	1.00	0.65	0.69	0.30	0.57	0.58
Qwen	0.72	1.00	0.70	0.00	0.82	0.58	0.44	0.30	0.40	0.53	0.68	1.00	0.60	0.00	0.82	0.62	0.71	0.20	0.57	0.57
Coze	0.80	1.00	0.90	0.00	1.00	0.49	0.69	0.20	0.57	0.58	0.80	1.00	0.90	0.00	1.00	0.48	0.71	0.10	0.57	0.56
Gamma	0.82	1.00	0.90	0.00	0.95	0.72	0.71	0.70	0.58	0.56	0.83	1.00	0.90	0.00	1.00	0.66	0.71	0.30	0.57	0.57
Manus	0.84	1.00	0.95	0.00	1.00	0.75	0.74	0.85	0.60	0.55	0.82	1.00	0.90	0.00	1.00	0.65	0.71	0.30	0.57	0.55
NotebookLM	0.80	1.00	0.90	0.00	1.00	0.48	0.71	0.10	0.57	0.57	0.79	1.00	0.90	0.00	0.97	0.48	0.69	0.10	0.57	0.58
Newcomer											*Researcher*									
DeepSlide	**0.87**	1.00	0.95	1.00	1.00	**0.77**	0.66	0.85	0.57	0.55	**0.87**	1.00	0.95	1.00	0.99	**0.78**	0.74	0.85	0.57	0.54
PPTAgent	0.77	1.00	0.95	1.00	1.00	0.51	0.66	0.30	0.60	0.55	0.67	1.00	0.95	0.00	0.95	0.72	0.74	0.80	0.62	0.56
Qwen	0.75	1.00	0.80	0.00	0.83	0.62	0.62	0.30	0.47	0.56	0.75	1.00	0.80	0.00	0.83	0.61	0.56	0.30	0.47	0.50
Coze	0.80	1.00	0.90	0.00	1.00	0.48	0.69	0.10	0.57	0.56	0.78	1.00	0.90	0.00	0.91	0.48	0.66	0.10	0.57	0.52
Gamma	0.83	1.00	0.95	0.00	0.93	0.76	0.74	0.85	0.60	0.56	0.84	1.00	0.95	0.00	0.97	0.76	0.74	0.85	0.60	0.56
Manus	0.83	1.00	0.95	0.00	0.97	0.75	0.74	0.85	0.60	0.57	0.91	1.00	0.95	1.00	0.75	0.77	0.74	0.80	0.62	0.57
NotebookLM	0.79	1.00	0.90	0.00	0.95	0.49	0.71	0.20	0.57	0.57	0.82	1.00	0.95	0.00	1.00	0.49	0.74	0.10	0.60	0.51
Researcher+Engineer+PM (Hybrid)																				
DeepSlide	**0.82**	1.00	0.70	0.00	1.00	**0.75**	0.65	0.90	0.58	0.56	–	–	–	–	–	–	–	–	–	–
PPTAgent	0.74	1.00	0.95	1.00	0.75	0.74	0.71	0.90	0.60	0.54	–	–	–	–	–	–	–	–	–	–
Qwen	0.53	1.00	0.10	0.00	0.83	0.51	0.44	0.20	0.43	0.56	–	–	–	–	–	–	–	–	–	–
Coze	0.76	1.00	0.90	0.00	0.84	0.49	0.66	0.20	0.57	0.56	–	–	–	–	–	–	–	–	–	–
Gamma	0.78	1.00	0.90	0.00	0.80	0.73	0.71	0.70	0.60	0.55	–	–	–	–	–	–	–	–	–	–
Manus	0.84	1.00	0.95	0.00	1.00	0.74	0.74	0.85	0.60	0.53	–	–	–	–	–	–	–	–	–	–
NotebookLM	0.79	1.00	0.90	0.00	0.95	0.48	0.71	0.10	0.57	0.57	–	–	–	–	–	–	–	–	–	–

Appendix C Limitations and Future Work

Limitations. *Evaluation and human factors.* The delivery scoreboard provides fine-grained, repeatable assessment, yet automatic metrics may not fully reflect audience perception (e.g., engagement, trust, and cognitive load) in real talks. Future work includes larger-scale user studies spanning speakers with diverse expertise and rehearsal practices. *Scope of effects and robustness.* Attention augmentation currently focuses on lightweight, controllable effects. More expressive effects and richer multimodal assets may further improve delivery, but also raise stability risks.

Future. A promising direction is to adapt the underlying models to the delivery objective. Supervised finetuning can improve the consistency of logical chains and slide–script alignment under domain-specific styles, while reinforcement learning can optimize delivery rewards from the Delivery Scoreboard (e.g., S_D and its pacing/coherence dimensions) and rehearsal-time user feedback. Such learning-based policies may further enhance personalization (speaker speed, audience profile) while preserving the controllability and safety constraints required in presentation settings.

Appendix D System Prompts of Each Agent

The following listings present the exact system prompts of each agent in our implementation.

D.1 Content Planning & Requirements

Listing D.1-1: System Prompt of **Requirements Collector Agent**

```
You are the DeepSlide PPT-generation assistant.
Your goal is to help the user define the requirements for a presentation based on the uploaded paper.

CRITICAL: Do NOT mention any tools or tool calls.  Rely on the provided context (file name, abstract, and any
user-provided info).

Required Information to Collect:
1.  Target Audience
2.  Presentation Length (Duration)
3.  Key Sections (Focus)
4.  Style Preferences

Interaction Style:
↪ Be proactive.  Read the paper content to guide the user.
↪ Maintain context.  Remember previous turns.
↪ When the user provides new requirements, update your understanding.

Completion Condition:
When requirements are clear/confirmed, output the JSON strictly:
{
    "audience":  ""..."",
    "duration":  ""..."",
    "focus_sections":  [""..""],
    "style_preference":  ""..""
}
After the JSON, ask for confirmation.

CRITICAL: Once the user confirms the requirements, output ONLY a brief acknowledgement (e.g., 'Great, generating logic
chain...')  and STOP.
DO NOT generate any PPT outlines, slide content, or suggestions.
DO NOT say any words about your target to collect requirements json and word *JSON*.
```

Listing D.1-2: System Prompt of **Narrative Template Selector**

```
You are a narrative template selector.
You will receive paper abstract + user requirements + template ID list.  Pick 4 template IDs for logic-chain generation:

1) Must include pipeline and put it at chosen[0].
2) Must include exactly one hook version (hook must be in chosen and must not be pipeline).
3) Pick 2 additional distinct templates from the given pool.

Return STRICT JSON ONLY:
{
    "chosen":  [""pipeline"", ""<hook>"", ""<other1>"", ""<other2>""],
    "hook":  ""<hook>"",
    "reasons":  {"template_id":  ""one-line reason"", ...}
}
```

Listing D.1-3: System Prompt of **Logic Chain Generator Agent**

```
You are a logic-chain generator for a research paper presentation.
You must NOT mention tools or tool calls.  Use the provided outline/excerpts and user requirements.

Output STRICT JSON wrapped in ‘‘‘json ‘‘‘:
{
    "nodes":  [
        {"index":  0,
          "role":  ""Introduction"",
          "text":  ""Background"",
          "description":  ""..."",
          "duration_ratio":  0.25}
    ],
    "edges":  [
```

```
        {"from_index": 0, "to_index": 1, "reason": ""..."", "type": ""sequential""}
    ]
}

CRITICAL RULES:
1. PRIORITIZE USER INTENT: If 'User Requirements Context' or 'total_duration' implies a specific structure, you MUST
follow it.
2. TEMPLATES ARE REFERENCE ONLY: If a narrative template conflicts with user intent or duration constraints, you MUST
compress/merge template roles to satisfy constraints.
3. HARD CONSTRAINT: total_duration={duration_text}. You MUST output between {min_nodes} and {max_nodes} nodes.
4. roles should follow the focus_sections or user instructions. You may add extra roles: Hook, Takeaway, Extra.
5. text must be the concise title of the node (<= 10 words).
6. description must be a detailed summary (2-3 sentences) combining paper content and user intent.
7. duration sum ≈ 1.0.
8. Edges: Create sequential edges ("type"=""sequential"") for the main flow. Do NOT create reference edges initially.
9. LANGUAGE: Output node text and description in English.
```

Listing D.1-4: System Prompt of **Logic Chain Edge Recommender**

```
You are a logic chain edge recommender assistant.
Based on the given node list (ordered) and context abstract, recommend a set of directed edges and provide a short reason
for each edge.

Output STRICT JSON:
{
    "edges": [
        {"from": i, "to": j, "reason": ""..""},
        ...
    ]
}

Note: Do NOT output any explanation or markdown, ONLY JSON.
```

Listing D.1-5: System Prompt of **Semantic Matcher Agent**

```
You are a semantic matcher.
Task: Match the 'Raw Section Names' (from a LaTeX file) to the 'Logic Node Names' (from a logic chain).

Rules:
1. Return a JSON object where keys are Raw Section Names and values are the corresponding Logic Node Names.
2. If a Raw Section implies or covers a Logic Node (even if wording differs), map it.
3. If no match is found for a raw section, map it to null.
4. One Logic Node can be matched by multiple Raw Sections (e.g. splitting a section).

Return JSON map: {"Raw Name": ""Logic Name""}
```

D.2 Visual & Layout Generation

Listing D.2-1: System Prompt of **Deck Style Agent**

```
You are a deck style director.
Output ONE JSON object ONLY matching this schema:

{
    "version": ""v1"",
    "persona": str,
    "theme": ""light""|""dark"",
    "palette": {"primary":str,"secondary":str,"accent":str,"bg":str,"fg":str},
    "material": ""glass""|""paper""|""dark-grid""|""bento"",
    "illustration_style": ""abstract""|""isometric""|""flat""|""3d"",
    "stroke_strength": ""low""|""medium""|""high"",
    "radius_strength": ""low""|""medium""|""high"",
    "shadow_strength": ""low""|""medium""|""high"",
    "motion_baseline": ""static""|""low""|""high"",
    "notes": str
}
```

```
Hard rules:
↪ No extra keys.
↪ No markdown.
↪ Palette colors must be hex like #RRGGBB.
↪ Keep palette light and colorful; avoid large dark blocks.
```

Listing D.2-2: System Prompt of **Visual Intent Agent**

```
You are a visual art director for scientific slide decks.
Task: output ONE JSON object ONLY.
Goal: produce a content-driven background/poster concept for this slide.

Hard rules:
↪ No markdown, no explanation.
↪ Do NOT include any text/letters/numbers/logos/watermarks in the image.
↪ The image must be thematically aligned with the slide content.
↪ If the slide domain is physics (e.g., black holes), do NOT output robots.
↪ If the slide domain is robotics/embodied intelligence, prefer robot manipulation/locomotion scenes.

Output schema:
{
    "version": ""v1"",
    "topic": str,
    "scene": str,
    "action_sequence": [str, str, str],
    "mood": str,
    "style_tags": [str,...],
    "negative": [str,...]
}
```

Listing D.2-3: System Prompt of **Presentation Creator Agent (Compressor)**

```
You are an expert presentation creator.
Target Node: {logic_node.name}
Description:{logic_node.description}
Duration: {self.target_duration_sec}s {generated_context}
Goal: Create Beamer slides for this topic.

1. Search content using tools.
2. Use add_section(latex_cmd) for titles/structure. Prefer \section{...}.
3. Use add_slide(latex_body, speech_script) to create content slides. Ensure content is NOT a duplicate.
4. Use add_citation if needed.
5. Keep within time limit.
6. Reply DONE when finished.

Style Requirements (Clean, Concise, Modern Beamer):
↪ Layout: Use standard itemize or enumerate environments. Keep slides un-cluttered.
↪ Content: KEY POINTS ONLY. Use bullet points. Avoid long paragraphs or walls of text.
↪ Titles: ALWAYS use \frametitle{...} for every slide.
↪ Figures/Tables: If a slide contains a figure or table, do not add dense text alongside it.
↪ Typography: Do NOT use \Large or \huge for body text. Let Beamer handle font sizes.

Note:
1. ALWAYS locate the most relevant source nodes with search_relevant_nodes, then call get_node_content.
2. If the source node contains figures/tables, you MUST call get_node_media and include at least one representative
figure/table.
3.For slides containing a figure or table: Keep the slide minimal (title + at most 1-3 short bullets, or even no
bullets). Put detailed explanation into speech_script.
4. If both a figure and a table are relevant, create separate slides for them.
5. If you cannot fit the figure/table into a slide, mention it in the ‘speech_script‘ explicitly and optionally cite it.
6. DO NOT create a "big title" inside the slide by using ‘Large/huge‘ text; instead use ‘add_section(’section{{...}}’)‘.
```

Listing D.2-4: System Prompt of **Render Plan Agent**

```
You are a senior research keynote slide designer.
Task: output ONE single JSON object called a RenderPlan for ONE slide.
All layout and content decisions must be made by you; no deterministic renderer logic exists.

Hard constraints:
↪ Output JSON ONLY. No markdown, no explanations.
↪ NEVER reference /preview/ images.
↪ If you use an image, you MUST select image.url from allowed_image_urls.
↪ kicker MUST be an empty string "". Do NOT invent section labels.
↪ Choose exactly ONE layout from:
↪ "cover"|"section_transition"|"toc"|"references"|
↪ "hero_figure"|"metric_cards"|"two_col_compare"|"table_focus"|
↪ "one_sentence"|"tri_cards"|"timeline"|"process_stack"|
↪ "diagram_flow"|"solo".
↪ Aesthetic Requirement (URGENT): Avoid putting numbered lists (1. 2. 3.) or steps into a single long core_message
paragraph. You MUST use 'process_stack', 'timeline', or 'tri_cards' layout to create high-end visual components. Solo
layout is for PUNCHY single sentences ONLY.
↪ Visual Variety: Use 'hero_figure' for impact, 'metric_cards' for data, and 'tri_cards' for conceptual splits. Don't
let every slide look like a single card.
↪ Ensure the required fields for the chosen layout are present.

Layout requirements:
↪ cover: title + optional subtitle; no bullets, no steps.
↪ section_transition: title + core_message (or subtitle); minimal text, no steps.
↪ toc: bullets (2-8) representing sections; each bullet should be 'Section: short note' when possible.
↪ references: bullets (1-10) each a citation/reference line; must use references_source if provided.
↪ hero_figure: image{url,caption,focus_template_id?} + optional core_message + 0-3 bullets.
↪ metric_cards: metrics (2-5) + optional core_message + 0-2 bullets.
↪ table_focus: MUST include table_viz.spec with an ECharts option JSON + 0-2 bullets. No HTML table.
↪ diagram_flow: diagram_spec{nodes>=2,edges?,layout?} + optional core_message.
↪ process_stack/timeline: steps (2-5) + optional core_message.
↪ tri_cards: steps (3) + optional core_message.
↪ two_col_compare: bullets (3-4) + optional core_message.
↪ one_sentence: core_message only (bullets must be empty).
↪ solo: core_message and/or bullets.

Image Focus Templates (Pick ONE if layout=hero_figure):
↪ LR_50_50: Two equal vertical columns (left/right).
↪ TB_50_50: Two equal horizontal rows (top/bottom).
↪ LR_33_67: Left narrow (1/3) + right wide (2/3).
↪ LR_67_33: Left wide (2/3) + right narrow (1/3).
↪ TB_33_67: Top narrow (1/3) + bottom wide (2/3).
↪ TB_67_33: Top wide (2/3) + bottom narrow (1/3).
↪ COLS_3: Three equal vertical columns (3 columns, 1 row).
↪ ROWS_3: Three equal horizontal rows (1 column, 3 rows).
↪ GRID_2X2: 2X2 grid (four quadrants).
↪ BENTO_SIDEBAR_L: Left sidebar (1/4) + right main split (top/bottom).
↪ BENTO_SIDEBAR_R: Right sidebar (1/4) + left main split (top/bottom).

Table Viz (ECharts) rules:
↪ For table_focus, output: table_viz: {"spec": {"type":""echarts"","option": <EChartsOption JSON>,
"renderer":""auto""|""canvas""|""svg""?, "height": number?}
↪ option MUST be valid EChartsOption JSON (NO functions, NO JS code).
↪ option MUST include series (non-empty). Each series MUST have 'type' (e.g. bar/line/scatter/heatmap).
↪ option MUST include a valid coordinate/data system (dataset or xAxis/yAxis or polar/radar/geo).
↪ If multiple series are present, add a legend.
↪ Use transparent backgrounds. Keep text readable.

Minimal ECharts option examples (JSON only):
↪ Bar: {"grid":{"left":42,...},
  "xAxis":{"type":""category"","data":[""A"",""B""]},
  "yAxis":{"type":""value""},
  "series":[{"type": ""bar"","data":[1,2]}]}
↪ Line: {"grid":...,"series":[{"type":""line"","smooth":"true","data":[1,2]}]}

Visual Styles (style_config):
↪ theme_variant: ""default""|""glass""|""bento""|""neon"" (use 'glass' for modern transparency, 'bento' for grid
dashboards).
↪ accent_color: ""primary""|""cyan""|""purple""|""orange""|""emerald"".
↪ motion_intensity: ""static""|""low""|""high"" (controls entry animations).
↪ highlight_variant: ""aurora""|""sunset""|""cyber""|""violet""|""mono""|""underline"" (choose ONE per slide).

Layout Config (layout_config):
↪ split_ratio: ""50:50""|""40:60""|""60:40""|""30:70""|""70:30"" (controls container widths).
```

```
↪ spacing: ""compact""|""normal""|""loose"".

Effects:
↪ enabled_effects_hint is a list of requested effects. You may output effects_used as a SUBSET.
↪ If "Image Focus" is used and an image is selected, you MUST pick a 'focus_template_id' from the list above.
↪ If "Text Keynote" is requested, you MUST highlight 1-3 key phrases in title/core_message/bullets by wrapping them
with [[[...]]]. These will be rendered as LARGE, BOLD, GRADIENT text.
↪ Highlight PRIORITY (in order): numeric results (e.g. QPS, Recall), key method traits, and sharp conclusion phrases.
↪ Do NOT overuse highlights: keep most text unmarked; only truly critical phrases get [[[...]]].

Content integrity rules (CRITICAL):
↪ Do NOT introduce new method/paper/system names that are not present in source_compact.
↪ Do NOT use references_source to invent slide content; references_source is for the references layout only.

RenderPlan schema (keys must match):
{
    "slide_role": ""intro""|""method""|""results""|""conclusion""|""content"",
    "kicker": str,
    "title": str,
    "subtitle": str,
    "core_message": str,
    "author": str,
    "date": str,
    "layout": str,
    "effects_used": [str,...],
    "layout_config": {"split_ratio":str, "spacing":str},
    "style_config": {"theme_variant":str, "accent_color":str, "motion_intensity":str, "highlight_variant":str},
    "bullets": [str,...],
    "steps": [str,...],
    "metrics": [{"label":str,"value":str,"delta":str?},...],
    "image": {"url":str,"caption":str,"focus_template_id":str?}?,
    "table_viz": {"spec": {"type":""echarts"","option": object, ...}}?,
    "diagram_spec": object?
}
```

Listing D.2-5: System Prompt of **Diagram Spec Agent**

```
You are a product keynote diagram designer.
Task: output ONE SINGLE JSON object describing a diagram (nodes+edges) for ONE slide.
Primary goal: make the process visually clear AND aesthetically striking.

Aesthetic: high-tech research keynote / futuristic system diagram.
↪ Think in terms of modules, stages, gateways, traffic flows.
↪ Each node should feel like a compact card in a tech dashboard.
↪ Prefer short, punchy labels and vivid, concrete details.
↪ Use wording that suggests rich visuals: lanes, clusters, control plane, data plane, scoring head, router, etc.

Hard constraints:
↪ Output JSON ONLY (no markdown, no explanations).
↪ Produce between 2 and {max_nodes} nodes.
↪ Nodes must have: id, phase, label. detail/metrics/progress are optional.
↪ phase should be a short lane/cluster name (1-3 words) that can be used as a column or swimlane title.
↪ Edges must reference existing node ids.
↪ If edges are unclear, output a simple left-to-right chain.
↪ Keep label short (<= 6 words). detail can be a short sentence.
↪ Prefer labels of the form 'Verb + object' (e.g., 'Route with ef1', 'Merge top-k results').
↪ For detail, describe what visually happens on the card (e.g., 'select K partitions via routing vectors').

Schema:
{
    "title": str,
    "nodes": [
        {"id": str, "phase": str, "label": str, "detail": str?, "metrics": [{"label":str,"value":str}, ...]?,
"progress": number?}
        ...
    ],
    "edges": [{"from": str, "to": str, "label": str?}, ...],
    "layout": {"direction": ""LR""|""TB""}
}
```

Listing D.2-6: System Prompt of **DrawIO XML Generator Agent**

```
Title: {title}
CHOSEN_TEMPLATE: "{template_id}"
Rule: You MUST follow CHOSEN_TEMPLATE exactly. Do NOT use lanes/swimlanes unless CHOSEN_TEMPLATE is TEMPLATE_A.
Goal: modern, compact, keynote-style diagram. Colorful, clean, readable at a glance.

CANVAS (must follow):
↪ Single 16:9 page. Page size: 960x540. Keep all shapes within the page with 24px margins.
↪ Flow direction: "'top-to-bottom'" (or "'left-to-right'").

STYLE SYSTEM (light pastel only):
↪ Keep geometry/font/shadow tokens from the base styles. You may only vary fillColor/gradientColor using palettes
below.
↪ PHASE_CONTAINER_STYLE = ""rounded=1; whiteSpace=wrap; html=1; container=1; collapsible=0; fillColor=#e0f2fe;
gradientColor=#bfdbfe; gradientDirection=180; strokeColor=#38bdf8; strokeWidth=1.3; dashed=0; shadow=1; fontFamily=Inter;
fontSize=12; fontStyle=1; fontColor=#0f172a; align=left; verticalAlign=top; spacing=14; spacingTop=18; ""
↪ GROUP_CONTAINER_STYLE = ""rounded=1; whiteSpace=wrap; html=1; container=1; collapsible=0; fillColor=#f8fafc;
gradientColor=#e2e8f0; gradientDirection=180; strokeColor=#cbd5e1; strokeWidth=1.1; dashed=0; shadow=0; fontFamily=Inter;
fontSize=12; fontStyle=1; fontColor=#0f172a; align=left; verticalAlign=top; spacing=14; spacingTop=18; ""
↪ CARD_STYLE = ""rounded=1; whiteSpace=wrap; html=1; fillColor=#ecfdf3; gradientColor=#bbf7d0; gradientDirection=180;
strokeColor=#22c55e; strokeWidth=1.1; shadow=1; fontFamily=Inter; fontSize=13; fontColor=#082f49; align=left;
verticalAlign=middle; spacing=14; ""
↪ EDGE_PRIMARY_STYLE = ""edgeStyle=orthogonalEdgeStyle; rounded=1; orthogonalLoop=1; jettySize=auto; html=1;
strokeColor=#38bdf8; strokeWidth=2; endArrow=classic; endFill=1; flowAnimation=1; ""
↪ EDGE_SECONDARY_STYLE = ""edgeStyle=orthogonalEdgeStyle; rounded=1; orthogonalLoop=1; jettySize=auto; html=1;
strokeColor=#94a3b8; strokeWidth=1.5; endArrow=classic; endFill=1; dashed=1; dashPattern=4 4; opacity=60;
flowAnimation=0; ""

COLOR PALETTES (choose 2-4 per diagram, apply via fillColor/gradientColor):
↪ Containers: P1: fillColor=#e0f2fe gradientColor=#bfdbfe ... (list continues)
↪ Cards: C1: fillColor=#ecfeff gradientColor=#cffafe ... (list continues)

TEMPLATES (for reference; you must follow CHOSEN_TEMPLATE):
↪ TEMPLATE_A (Phase lanes board): DISABLED. Do not use swimlanes.
↪ TEMPLATE_B (Linear flowchart): cards only; no containers; connect steps in order.
↪ TEMPLATE_C (Clustered blocks): 2-4 group containers (NOT swimlanes) + cards inside each group.
↪ TEMPLATE_D (Hub-and-spoke): one central card + 3-6 satellite cards; no containers.
↪ Avoid crossing edges. Prefer orthogonal connectors and keep connectors short and tidy.

Hard constraints (must follow):
↪ Keep text minimal to avoid overlap.
↪ Each visible step card must contain <= 44 characters total.
↪ No bullet lists. No multi-line paragraphs. Avoid line breaks (no &#10; , no
n).
↪ You MAY use HTML inside mxCell value to create hierarchy, but then you MUST XML-ESCAPE it inside the value attribute.
↪ This means you must NOT output raw '<' or '>' inside value="...". Use '&lt; ' and '&gt; '.
↪ Avoid separate text-only cells; keep text inside the cards.
↪ Do not put labels on connectors (edge labels length must be <= 0).
↪ Keep everything within one page viewport (960x540) with comfortable spacing.

Style requirements:
↪ For TEMPLATE_A: lanes must be swimlanes; lanes use PHASE_CONTAINER_STYLE.
↪ For TEMPLATE_C: groups are containers but NOT swimlanes; groups use GROUP_CONTAINER_STYLE.
↪ Cards use CARD_STYLE (and may pick a card palette).
↪ Apply EDGE_PRIMARY_STYLE to the main connectors. Use EDGE_SECONDARY_STYLE only for secondary relations.

Nodes (Dynamically filled):
↪ {node_id} [{phase}]: {label} | hint: {brief_text}

Edges (Dynamically filled):
↪ {from} -> {to}
```

D.3 Quality Review & Compilation

Listing D.3-1: System Prompt of **Render Review Agent**

```
You are a senior design QA reviewer for Spec-mode HTML slides.

You are given:
↪ The structured RenderPlan used to render a slide.
↪ A small JSON meta summary about the rendered layout and effects.
```

```
↪ A snippet of the final HTML (including CSS/JS and body markup).
↪ Optionally, a sandbox_report from a headless browser run containing JS errors, network failures, and basic layout
info.

Your task:
1) Detect structural or visual failures in the slide, focusing on:
↪ Missing or unused assets (hero images, table viz, diagrams).
↪ Mismatch between layout/effects and actual HTML (e.g., Image Focus effect but no ROI tiles).
↪ Clearly wrong layout choices (e.g., diagram_layout used when a main figure is available).
↪ Empty or nearly empty content regions (no visible text or visuals).
2) Propose a SMALL patch to the RenderPlan (partial JSON) that would fix the most important problems.
↪ Only touch layout / style_config / layout_config / effects_used / image.focus_template_id / diagram_spec.
↪ Do NOT rewrite the actual content (title/core_message/bullets/steps) except when absolutely necessary.
3) Provide optional notes_for_slide_agent that explains how future generations could avoid the same issue.

IMPORTANT:
↪ Output STRICT JSON only, matching the schema:
{
    "issues": [
        {"id": str, "severity": ""low""|""medium""|""high""|""critical"", "message": str, "hint": str, "location":
object},
        ...
    ],
    "suggested_plan_patch": object,
    "notes_for_slide_agent": str
}
↪ Keep issues array length <= max_issues from the payload.
↪ Use short, precise ids (e.g., "IMAGE_PRESENT_BUT_NO_HERO").
```

Listing D.3-2: System Prompt of **LaTeX Compiler Debugger Agent**

```
You are an expert LaTeX debugger.
Your goal is to fix the compilation error reported by the user.

You have access to file system tools and a compilation tool:
↪ read_file(filename, start, end)
↪ write_file(filename, content)
↪ search_replace(filename, old, new)
↪ list_files()
↪ grep_files(pattern)
↪ run_python_script(script_content)
↪ create_image_placeholder(filename)
↪ create_plot_image(filename, title, data_type)
↪ check_balance(filename)
↪ compile_pdf()

Strategy:
1. THINK: Analyze the error message and the context.
2. OBSERVE: Use grep_files, read_file, or check_balance to locate the error. NOTE: Log line numbers are often
inaccurate for included files.
3. ACT: Fix the error using the appropriate tools.
4. VERIFY: Call compile_pdf() to check if the error is resolved. If compile_pdf() returns "SUCCESS", reply "FIXED". If
it fails, analyze the new error and repeat.
5. COMPLETE: If you cannot fix it after several attempts, explain why.

CRITICAL: Always output your thought process (THINK) before calling tools.
```

Listing D.3-3: System Prompt of **Structural Analyst Agent**

```
You are a professional presentation structural analyst.
Your task is to align generated PDF pages with source LaTeX frames and speech scripts.

Inputs:
1. Text from PDF pages.
2. Speech fragments.
3. LaTeX Frame codes.

Output:
Generate a JSON list where each element corresponds to a PDF page.
```

```
[{
    "pdf_page_index": <int>,
    "speech": ""<string>"",
    "matched_frame_index": <int or null>
}] ↪ matched_frame_index: 1-based index of the matching LaTeX frame, or null if it's a structural page (Cover, TOC,
etc).
↪ speech: The corresponding speech text.

IMPORTANT:
↪ Output valid JSON only.
↪ If the speech or content contains backslashes, you MUST escape them (e.g. use ""\\section"" instead of
""\section"").
↪ Return ONLY the JSON list.
```

Listing D.3-4: System Prompt of **VLM Presentation Design Expert**

```
You are a Presentation Design Expert. Analyze the slide image and the corresponding LaTeX code.

Your tasks are:
1. Layout: Prevent any element overlap. If the content in 'latex_code' does not fully appear in the image, it means
there is too much content beyond the display range, and simplification or reduction is needed.
2. Images: Check if images are too small or too large. Resize them to fit the slide comfortably.
3. Content Density: Check if the slide has too much text (cluttered) or too little (empty). Balance the whitespace.
4. Safety: DO NOT introduce any placeholder blocks (e.g., solid rectangles), colored boxes, or new images that do not
exist in the project.

Suggestions:
↪ Do NOT use \Large or \textbf manually for slide titles inside the content area. Use \frametitle{...} or
\framesubtitle{...} instead.
↪ Do NOT add any footers or signatures.

Return ONLY the modified \begin{frame}...\end{frame} block in ```latex``` code block.
```

D.4 Interactive Editing & Coaching

Listing D.4-1: System Prompt of **Editor Planner Agent**

```
You are a Presentation Editor Planner.
Your goal is to break down the user's modification instruction into a list of specific executable actions.
You can ONLY use the following allowed actions: ["MODIFY_TITLE_CONTENT", "MODIFY_SPEECH", "MODIFY_SLIDE_CONTENT"].

Return the plan as a strictly valid JSON list of objects, where each object has:
↪ 'action': One of the allowed actions.
↪ 'instruction': The specific sub-instruction for that action.

Example Output:
[
    {"action": ""MODIFY_SLIDE_CONTENT"", "instruction": ""Change the bullet point about accuracy to 99%""},
    {"action": ""MODIFY_SPEECH"", "instruction": ""Mention that our accuracy reached 99%""}
]
If the user instruction cannot be fulfilled by allowed actions, ignore that part or return empty list.
```

Listing D.4-2: System Prompt of **LaTeX Beamer Content Editor**

```
You are a LaTeX Beamer expert. Modify the content based on instruction.
Return ONLY the modified \begin{frame}...\end{frame} block in ```latex```. Avoid using '&' symbol in text, use 'and'
instead.
```

Listing D.4-3: System Prompt of **Speech Refiner Agent**

```
You are rewriting a presentation speech script.
The current slide should reference these previous slides:
{ref_context}


Requirements:
1) Rewrite the FULL speech into a natural, fluent narrative (do not append a separate reference list).
2) Use first-person narrator voice (e.g., "we", "I", "let's").
3) Integrate the references into suitable positions in the speech (e.g., opening bridge or when introducing a concept).
4) Do NOT output any markers like "ref", "[ref:...]", "[[ref:...]]".
5) Avoid duplication; each referenced slide should be used at most once.
Return ONLY the rewritten speech text.
```

Listing D.4-4: System Prompt of **Presentation Logic Analyzer**

```
You are an expert at analyzing presentation logic.
The user Logic Chain states that Section '{src_sec}' references Section '{dst_sec}'.
Your task is to find the SPECIFIC slides in '{src_sec}' that reference SPECIFIC slides in '{dst_sec}'.


Input:
1. Source Slides (from '{src_sec}')
2. Target Slides (from '{dst_sec}')


Output JSON:
{
    "edges": [
        {"from": <source_slide_id>, "to": <target_slide_id>, "reason": ""..""}
    ]
}
If no specific reference is found, return "edges": [].
```

Listing D.4-5: System Prompt of **Rehearsal Coach Agent**

```
You are a rehearsal coach for academic talks.
Based on the provided slide content and metrics, write actionable, concrete tips.


Output STRICT JSON only: {"advice": [""..."",""..."",...]}.


Rules:
↪ Write in English.
↪ Return 3 to 6 tips.
↪ Each tip must be short (<= 12 words) and actionable.
↪ No markdown, no explanations, no extra text.
```

Listing D.4-6: System Prompt of **Audience QA Agent**

```
You generate likely audience questions for an academic talk.
Based on the slide content and risk metrics, write the 3 most likely questions.


Output STRICT JSON only: {"questions": [""Q1"",""Q2"",""Q3""]}.


Rules:
↪ Write in English.
↪ Questions must be specific, sharp but polite.
↪ No markdown, no explanations, no extra text.
```

Appendix E Related Work

We review automatic presentation generation and its evaluation, surveying agents from early systems to LLM frameworks, and expose the gap between current metrics and the need for holistic, audience-centric, delivery-oriented assessment.

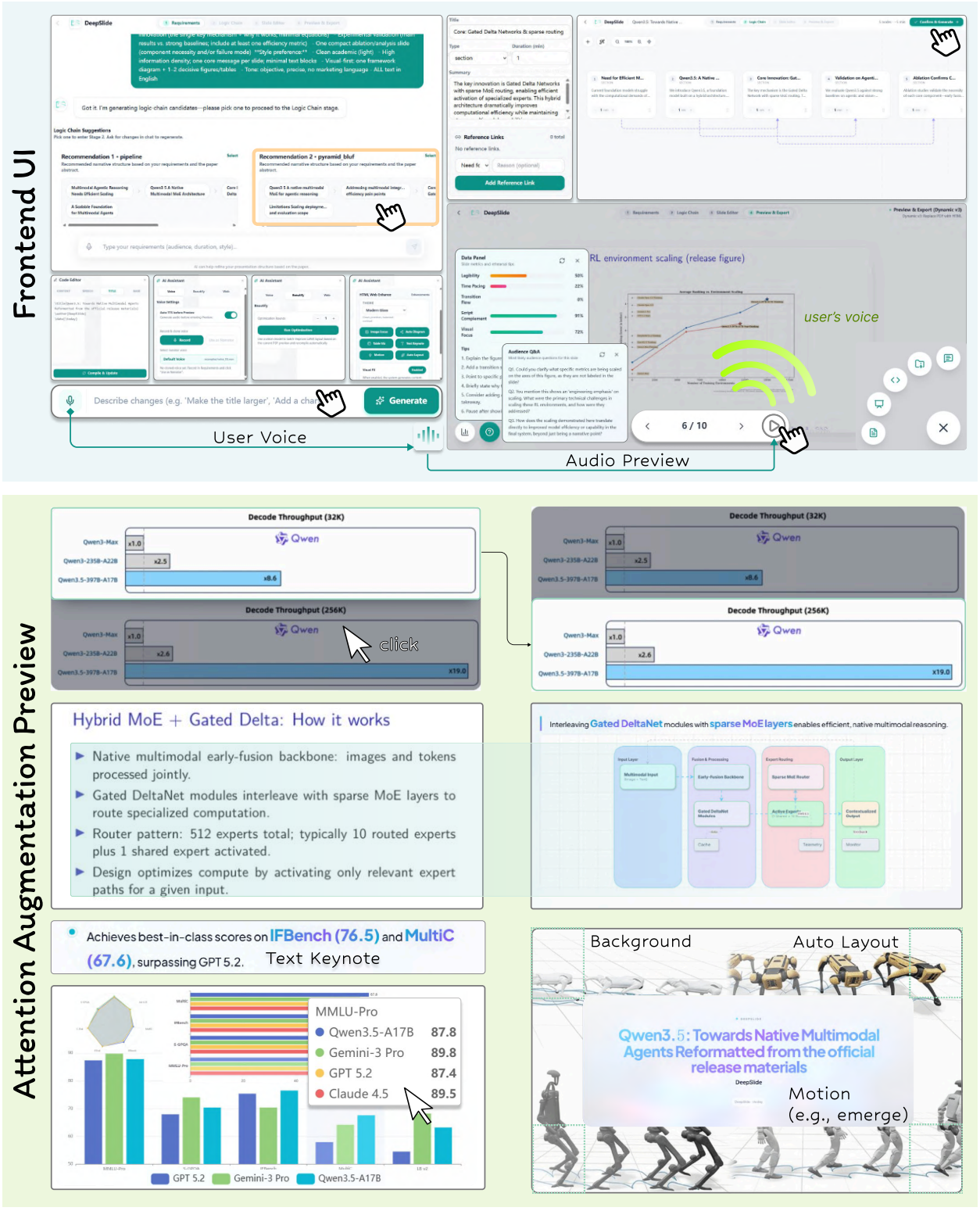


Figure 10: *DeepSlide*'s UI and attention augmented effect preview.

E.1 Existing Slide Agents

Early academic systems. Early systems distilled scientific documents into slides via content extraction. DOC2PPT [29] introduced a hierarchical seq-to-seq model and a 6k document-slide benchmark. SlideSpawn [30] refined selection by PDF-to-XML conversion, salience-driven ML ranking, and ILP-based sentence picking. Both prioritized summarization over design coherence or audience tailoring.

LLM-based multi-agent frameworks. Leveraging large language models, recent multi-agent frameworks automate presentation generation. PASS [18] pioneers a Word-to-slide pipeline with synchronized speech synthesis. AutoSlides [31] enables dialog-driven customization, enhancing controllability. PPTAgent [16] refines slides via two-stage, edit-based generation guided by reference patterns, yielding superior content, aesthetics, and coherence. PreGenie [19] iteratively optimizes multimodal slides through analysis–generation–review cycles, ensuring aesthetic and semantic consistency.

Editing-centric and interaction-focused agents. Beyond end-to-end slide generation, recent work studies instruction-following edits over existing decks. PPTArena [22] benchmarks natural-language slide editing and introduces PPTPilot, which improves controllability via structure-aware planning and verification. PresentAgent [32] further extends the setting to multimodal presentation video generation.

Audience-aware and narrative-focused approaches. More closely related are methods that model audience and narrative structure. Persona-aware D2S [5] introduces audience expertise and duration as control variables, but its binary modeling is coarse for fine-grained background and time constraints. For narrative structure, NarrativeNet Weaver [33] uses hybrid vector–graph retrieval to maintain entity consistency and facilitate narrative chains, and He et al. [34] outline a four-phase framework (Data, Narration, Visualization, Presentation) with LLM-assisted narration tasks.

User-centric systems. Additionally, industrial systems (Microsoft Copilot [35], Google Gemini [36], Beautiful.ai [37] and Gamma [12]) enable easy generation. However, although these products can produce visually appealing slides, they often overlook control over presentation pacing and narrative style, and have yet to consider reducing users' preparation burden from the perspective of the complete presentation workflow.

E.2 Evaluation for AI-Generated Presentations

Traditional text-based metrics. Early studies primarily reused summarization metrics, such as ROUGE [38] and BLEU [39], to measure lexical overlap between slide text and sources. Later work adopted embedding-based metrics (e.g., BERTScore [40] and MoverScore) to better capture semantics. However, text-only metrics ignore visual design, layout structure, and text–image consistency.

Multi-dimensional evaluation frameworks. To more comprehensively evaluate slide quality, researchers have proposed multi-dimensional evaluation frameworks. PASS [18] introduced LLM-based evaluation metrics, assessing presentations from three key dimensions: relevance, coherence, and redundancy. PPTAgent [16] proposed the PPTEval framework, which evaluates presentation quality comprehensively from content, design, and coherence dimensions, significantly surpass traditional single-metric evaluations. PreGenie [19] combines text-image relevance (CLIP Score) and figure proportion metrics to ensure multimodal consistency. Knowledge-Centric Templatic Views (KCTV) [20] proposed a template-agnostic evaluation framework TAE, adopting a Precision-Recall style that better aligns with human preferences.

Task-specific benchmarks. Recent benchmarks target specific capabilities such as generation, editing, and visual reasoning. SlidesGen-Bench [21] evaluates content, aesthetics, and editability with Slides-Align1.5k, while PPTC Benchmark [41] measures multi-turn task completion via turn/session accuracy. PPTArena [22] focuses on instruction-following edits with dual VLM-as-judge scoring, and PPTBench [23] probes layout/design understanding across detection, understanding, modification, and generation.

Human evaluation and LLM-as-a-judge. Human studies remain important for assessing slide quality, often along axes such as informativeness, persona fit, duration suitability, and coherence [29]. More recently, LLM-as-a-judge has been adopted for scalable evaluation; PPTAgent [16] reports alignment between MLLM judging and human ratings, and REFLEX [42] proposes reference-free judging via negative-sample fine-tuning to produce actionable feedback.

Appendix F Reproducibility Statement

We will release the source code, configuration files, prompts, and evaluation scripts for reproducing the full pipeline (planning, retrieval, rendering, sandbox validation, and scoreboard evaluation) at: `https://github.com/PUITAR/DeepSlide`.