PPTAGENT: Generating and Evaluating Presentations Beyond Text-to-Slides

Hao Zheng[1,2,*], **Xinyan Guan**[1,2,*], **Hao Kong**[3], **Wenkai Zhang**[1], **Jia Zheng**[1,†], **Weixiang Zhou**[1], **Hongyu Lin**[1,†], **Yaojie Lu**[1,†], **Xianpei Han**[1], **Le Sun**[1]
[1]Chinese Information Processing Laboratory, Institute of Software, Chinese Academy of Sciences, Beijing, China
[2]University of Chinese Academy of Sciences, Beijing, China
[3]Shanghai Jiexin Technology, Shanghai, China
{zhenghao2022,guanxinyan2022,zhengjia,hongyu,luyaojie}@iscas.ac.cn

Abstract

Automatically generating presentations from documents is a challenging task that requires accommodating content quality, visual appeal, and structural coherence. Existing methods primarily focus on improving and evaluating the content quality in isolation, overlooking visual appeal and structural coherence, which limits their practical applicability. To address these limitations, we propose PPTAGENT, which comprehensively improves presentation generation through a two-stage, edit-based approach inspired by human workflows. PPTAGENT first analyzes reference presentations to extract slide-level functional types and content schemas, then drafts an outline and iteratively generates editing actions based on selected reference slides to create new slides. To comprehensively evaluate the quality of generated presentations, we further introduce PPTEVAL, an evaluation framework that assesses presentations across three dimensions: Content, Design, and Coherence. Results demonstrate that PPTAGENT significantly outperforms existing automatic presentation generation methods across all three dimensions.[1]

1 Introduction

Presentations are a widely used medium for information delivery, valued for their visual effectiveness in engaging and communicating with audiences. However, creating high-quality presentations requires a captivating storyline, well-designed layouts, and rich, compelling content (Fu et al., 2022). Consequently, creating well-rounded presentations requires advanced presentation skills and significant effort. Given the inherent complexity of the presentation creation, there is growing interest in automating the presentation generation process (Ge et al., 2025; Maheshwari et al., 2024; Mondal et al., 2024) by leveraging the generalization capabilities of Large Language Models (LLMs) and Multimodal Large Language Models (MLLMs).

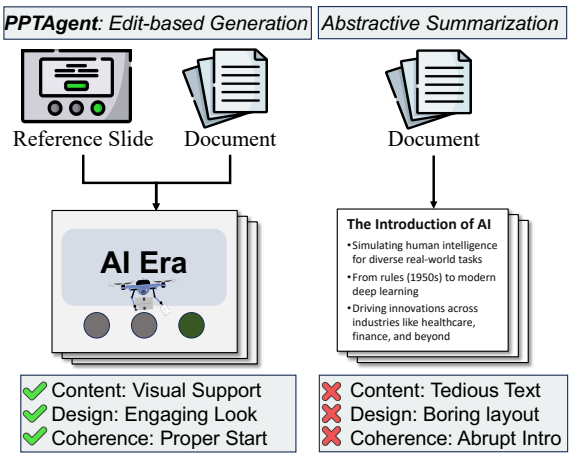

Figure 1: Comparison between our PPTAGENT approach (left) and the conventional abstractive summarization method (right).

Existing methods for presentation generation typically adhere to a text-to-slides paradigm (Mondal et al., 2024; Sefid et al., 2021), wherein generated text is converted into slides relying on a limited set of human-defined rules or templates. As illustrated in Figure 1, this approach often simplifies the task to an extension of abstractive summarization (Mondal et al., 2024; Sefid et al., 2021) and operates without holistic planning. Consequently, the resulting presentations are often text-heavy and fragmented, thus failing to engage audiences (Barrick et al., 2018), underscoring the necessity to broaden the scope and depth of aspects considered in presentation generation research.

Rather than creating complex presentations from scratch, human workflows typically involve selecting exemplary slides as references and then transferring key content onto them (Duarte, 2010). However, enabling LLMs to adopt this edit-based

[*]These authors contributed equally to this work.
[†]Corresponding authors.
[1]This project is available at https://github.com/icip-cas/PPTAgent

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 14402–14418
November 4-9, 2025

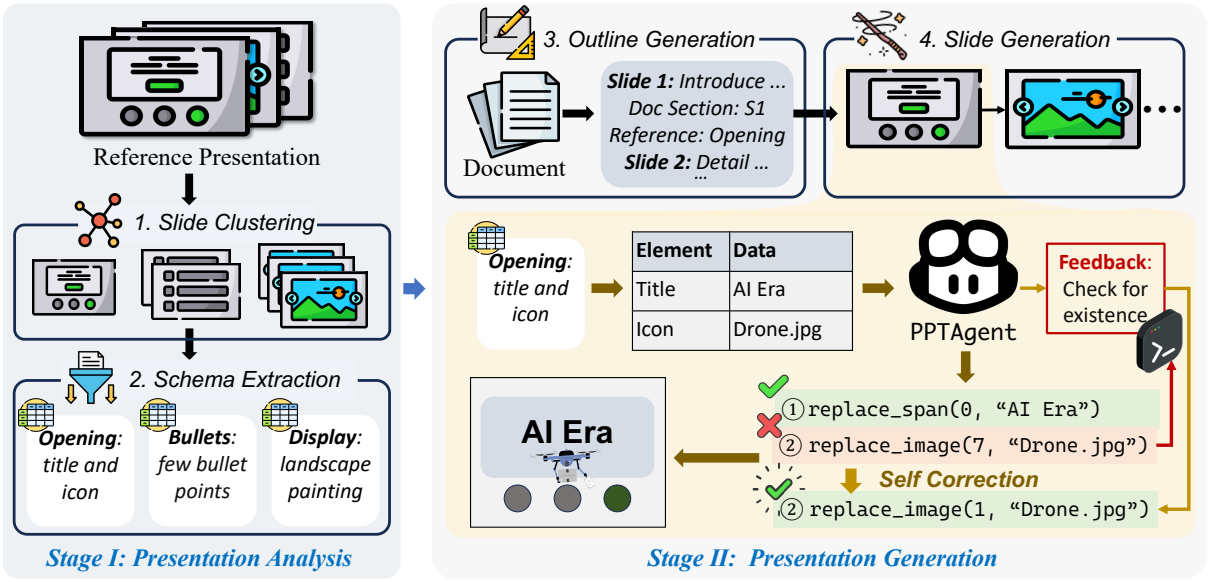

Figure 2: Overview of the PPTAGENT workflow. ***StageI: Presentation Analysis*** involves analyzing the reference presentation to cluster slides into groups and extract their content schemas. ***Stage II: Presentation Generation*** generates new presentations guided by the outline, incorporating self-correction mechanisms to ensure robustness.

paradigm for presentation generation poses several challenges. First, the inherent functional diversity and layout complexity of presentations make it difficult for LLMs to directly determine which slides should be referenced. Second, most presentations are saved in PowerPoint's verbose and redundant XML format (Gryk, 2022), as demonstrated in Figure 11, which hinders LLMs from performing robust editing operations. This gap between real-world demands for interactive presentation editing and the limitations of LLMs in understanding and manipulating presentations raises an intriguing question: *Can we devise an agentic workflow that achieves human-level effectiveness with the edit-based paradigm?*

In this work, we propose PPTAGENT, which addresses these challenges in two stages. In stage I, we analyze reference presentations to extract slide-level functional types (which classify slides by purpose or layout pattern) and their content schemas, facilitating subsequent reference selection and slide generation. To preserve rich details within slides and enable fine-grained modifications for handling presentation complexity (Wang et al., 2024b), we introduce a suite of APIs that operate on HTML-rendered slides. In Stage II, LLMs utilize these APIs to simplify slide modifications through code interaction. Moreover, we introduce a self-correction mechanism that allows LLMs to refine their output using execution failures as feedback, thereby ensuring the robustness of the generation process (Kamoi et al., 2024). As shown in Figure 2, we first cluster the reference slides into categories (e.g., "Opening") and extract their content schemas. In the next stage, the LLM plans the new slides by constructing a presentation outline that defines each one's purpose (e.g., Slide 1 to "Introduce the presentation topic"), content source, and reference slide. For each slide, PPTAGENT produces a series of editing actions (e.g., replace_span) to transfer the content, which is generated under the guidance of the content schema, onto the slide.

Due to the lack of a comprehensive evaluation framework, we propose PPTEVAL, which adopts the MLLM-as-a-judge paradigm (Chen et al., 2024a) to evaluate presentations across three dimensions: Content, Design, and Coherence(Duarte, 2010). Human evaluations validate the reliability and effectiveness of PPTEVAL. Results demonstrate that PPTAGENT generates high-quality presentations, achieving an average score of 3.67 for the three dimensions in PPTEVAL.

Our main contributions can be summarized as follows:

- We propose PPTAGENT, a framework that redefines automatic presentation generation as an edit-based process guided by reference presentations.
- We introduce PPTEVAL, a comprehensive evaluation framework that assesses presentations

across three dimensions: Content, Design, and Coherence.

- We release the PPTAGENT and PPTEVAL codebases, along with a new presentation dataset *Zenodo10K*, to support future research.

2 PPTAGENT

In this section, we formulate the presentation generation task and introduce our proposed PPTAGENT framework, which consists of two distinct stages. In stage I, we analyze reference presentations through slide clustering and schema extraction, providing a comprehensive understanding of reference presentations that facilitates subsequent reference selection and slide generation. In stage II, we leverage the comprehension of reference presentations to select reference slides and generate the target presentation for the input document through an iterative editing process. An overview of our workflow is illustrated in Figure 2.

2.1 Problem Formulation

PPTAGENT is designed to generate an engaging presentation through an edit-based process. We provide formal definitions for the conventional method and PPTAGENT to highlight their key differences.

Conventional methods (Bandyopadhyay et al., 2024; Mondal et al., 2024) for creating each slide $\boldsymbol{S}$ is formalized in Equation 1. Given the input content C, models generates n slide elements e_i, each defined by its type, content, and styling attributes, such as $(\text{Textbox}, \text{"Hello"}, \{\text{size}, \text{position}, \dots\})$.

$$\boldsymbol{S} = \{e_1, e_2, \dots, e_n\} = f(C) \tag{1}$$

While this conventional method is straightforward, it requires manual specification of styling attributes, which is challenging for automated generation (Guo et al., 2023). Instead of creating slides from scratch, PPTAGENT generates a sequence of executable actions to edit reference slides, thereby preserving their well-designed layouts and styles. As shown in Equation 2, given the input content C and the j-th reference slide R_j, which is selected from the reference presentation, PPTAGENT generates a sequence of m executable actions, where each action a_i corresponds to a line of executable code.

$$\boldsymbol{A} = \{a_1, a_2, \dots, a_m\} = g(C, R_j) \tag{2}$$

2.2 Stage I: Presentation Analysis

In this stage, we analyze the reference presentation to guide the reference selection and slide generation. Firstly, we categorize slides based on their structural and layout characteristics through slide clustering. Then, we model the content structure of slides within each cluster into a defined content schema, which provides a comprehensive description of its constituent elements.

Slide Clustering Slides can be categorized into two main types based on their functionalities: structural slides that support the presentation's organization (e.g., opening slides) and content slides that convey specific information (e.g., bullet-point slides). To distinguish between these two types, we employ LLMs to segment the presentation accordingly. For structural slides, we leverage LLMs' long-context capability to analyze all slides in the reference presentation, identifying structural slides, labeling their structural roles based on their textual features, and grouping them accordingly. For content slides, we first convert them into images and then apply a hierarchical clustering approach to group similar slide images. Subsequently, we utilize MLLMs to analyze the converted slide images, identifying layout patterns within each cluster. Further details are provided in Appendix D.

Schema Extraction After clustering, we further analyzed their content schemas to facilitate the slide generation. Specifically, we define an extraction framework where each element is represented by its category, description, and content. This framework enables a clear and structured representation of each slide. Detailed instructions are provided in Appendix F, with an example of the schema shown below.

Category	Description	Data
Title	Main title	Sample Library
Date	Date of the event	15 February 2018
Image	Primary image to illustrate the slide	Picture: Children in a library with ...

Table 1: Example of the extracted content schema.

2.3 Stage II: Presentation Generation

PPTAGENT first holistically plans by generating a detailed presentation outline. Guided by this, it then iteratively edits selected reference slides using the provided APIs to create the target presentation

Outline Generation As shown in Figure 2, we utilize LLMs to construct a structured presentation outline of multiple entries, with each entry specifying the new slide's purpose, its reference slide, and relevant content to be retrieved from the input document. This planning process utilizes the LLM's understanding of slide functionality (from Stage I) and its grasp of the input document's structure and available images (via captions). Such detailed upfront planning ensures generated slides are contextually appropriate, thereby contributing to the overall coherence of the presentation.

Slide Generation Guided by the presentation outline from the previous phase, slides are generated iteratively. For each new slide, the LLM leverages text and image captions retrieved from the input document to produce the content of the new slide, under the guidance of the content schema. Subsequently, the LLM transfers the content onto the selected reference slide, thus ensuring the generated slide adopts the layout of the reference slide while maintaining consistency in content structure.

Specifically, we provide a suite of APIs that enable LLMs to perform various editing operations on slides. These APIs support granular control over slide elements, allowing the LLM to edit text content, insert images and tables extracted from the input document, and remove or duplicate existing elements. Moreover, given that direct LLM interaction with verbose and complex presentation XML can be unreliable, we employ rule-based conversion to render reference slides into an HTML representation (demonstrated in Figure 10) for a more precise and intuitive structure. This LLM-friendlier HTML format (Feng et al., 2024), combined with our provided APIs, crucially enables accurate programmatic editing by the LLM.

Function Name	Description
`del_span`	Delete a span.
`del_image`	Delete an image element.
`clone_paragraph`	Clone an existing paragraph.
`replace_span`	Replace the content of a span.
`replace_image`	Replace the source of image.

Table 2: Definition and function of the provided APIs.

Furthermore, to enhance the robustness of the editing process, we implement a self-correction mechanism adapted from Kamoi et al. (2024). Specifically, we execute the LLM-generated editing code within a Python REPL[2] environment. When the generated code attempts invalid operations (e.g., editing non-existent slide elements), the Python interpreter captures these runtime errors[3] and returns detailed error messages. These error messages are then appended to the conversation history and fed back to the LLM as additional context. This feedback enables the LLM to understand what went wrong and generate corrected code accordingly (Guan et al., 2024; Wang et al., 2024b). This iterative refinement continues until valid slides are generated or the maximum retry limit is reached.

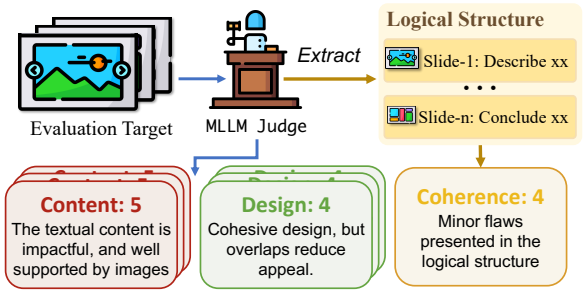

Figure 3: PPTEVAL assesses presentations from three dimensions: content, design, and coherence.

Dimension	Criteria
Content	Text should be concise and grammatically sound, supported by relevant images.
Design	Harmonious colors and proper layout ensure readability, while visual elements like geometric shapes enhance the overall appeal.
Coherence	Structure develops progressively, incorporating essential background information.

Table 3: The scoring criteria of dimensions in PPTEVAL, all evaluated in 1-5 scale.

3 PPTEVAL

We introduce PPTEVAL, a comprehensive framework that evaluates presentation quality from multiple dimensions, addressing the absence of label-free evaluation for presentations. The framework provides both numeric scores (1-to-5 scale) and detailed rationales to justify its assessment.

Grounded in established presentation design principles (Duarte, 2008, 2010), our evaluation framework focuses on three key dimensions, as summarized in Table 3. Specifically, given a presentation, we assess the content and design at the slide level, while evaluating coherence across the

[2] https://en.wikipedia.org/wiki/REPL
[3] https://docs.python.org/tutorial/errors

entire presentation. The complete evaluation process is illustrated in Figure 3, with representative examples, and details of scoring criteria and human agreement evaluation are provided in Appendix C.

4 Experiment

In this section, we conduct a series of experiments to answer the following research questions:

- **RQ1:** Does PPTAGENT, as an edit-based approach, outperform existing baselines?
- **RQ2:** Does PPTAGENT leverage the reference presentation effectively?
- **RQ3:** How can presentations be evaluated reliably and comprehensively?

4.1 Dataset

Existing presentation datasets, such as Fu et al. (2022); Mondal et al. (2024); Sefid et al. (2021); Sun et al. (2021), have two main issues. First, they are mostly stored in PDF or JSON formats, which leads to a loss of semantic information, such as structural relationships and styling attributes of elements. Furthermore, these datasets mainly consist of academic presentations on artificial intelligence, which limits their diversity. To address these limitations, we introduce *Zenodo10K*, a new dataset sourced from Zenodo (European Organization For Nuclear Research and OpenAIRE, 2013), which hosts various artifacts across domains, all under clear licenses. We have curated 10,448 presentations from this source and made them publicly available to support further research.

Following Mondal et al. (2024), from each of the five domains, we sampled 10 input documents (to serve as the source of content) and 10 reference presentations (to provide stylistic and structural guidance). This dataset composition yields 500 presentation generation tasks per experimental configuration (5 domains × 10 input documents × 10 reference presentations). Table 4 provides overall dataset statistics, with detailed sampling criteria and preprocessing steps provided in Appendix A

4.2 Implementation Details

PPTAGENT is implemented with three models: GPT-4o-2024-08-06 (GPT-4o), Qwen2.5-72B-Instruct (Qwen2.5, Yang et al., 2024), and Qwen2-VL-72B-Instruct (Qwen2-VL, Wang et al., 2024a). These models, categorized by the modality (textual or visual) they handle as indicated by subscripts, are further combined into configurations consisting of a language model (LM) and a vision model (VM), such as $\text{Qwen2.5}_{\text{LM}}$+$\text{Qwen2-VL}_{\text{VM}}$. Each slide generation allows a maximum of two self-correction iterations. We use Chen et al. (2024b) and Wu et al. (2020) to compute the text and image embeddings respectively. All open-source LLMs are deployed using the VLLM framework (Kwon et al., 2023) on NVIDIA A100 GPUs. The total computational cost for experiments is approximately 500 GPU hours.

Domain	*Document*		*Presentation*		
	#Chars	#Figs	#Chars	#Figs	#Pages
Culture	12,708	2.9	6,585	12.8	14.3
Education	12,305	5.5	3,993	12.9	13.9
Science	16,661	4.8	5,334	24.0	18.4
Society	13,019	7.3	3,723	9.8	12.9
Tech	18,315	11.4	5,325	12.9	16.8

Table 4: Statistics of the dataset used in our experiments, detailing the number of characters ('#Chars') and figures ('#Figs'), as well as the number of pages ('#Pages').

4.3 Baselines

We choose the following baseline methods:

DocPres (Bandyopadhyay et al., 2024) proposes a rule-based approach that generates narrative-rich slides through multi-stages, and incorporates images through a similarity-based mechanism.

KCTV (Cachola et al., 2024) proposes a template-based method that creates slides in an intermediate format before converting them into final presentations using predefined templates.

The baseline methods operate without vision models since they do not process visual information. Each configuration generates 50 presentations (5 domains × 10 input documents), as they require predefined templates instead of reference presentations. Consequently, the FID metric is excluded from their evaluation.

4.4 Evaluation Metrics

We evaluated the presentation generation using the following metrics:

- **Success Rate (SR)** evaluates the robustness of presentation generation (Wu et al., 2024), calculated as the percentage of successfully completed tasks. For PPTAGENT, success requires the generation of all slides without execution errors after self-correction. For KCTV, success is determined by the successful compilation of the generated LaTeX file. DocPres is excluded from this evaluation due to its deterministic rule-based conversion.

Configuration		Existing Metrics				PPTEVAL			
Language Model	Vision Model	SR(%)↑	PPL↓	ROUGE-L ↑	FID↓	Content↑	Design↑	Coherence↑	Avg.↑
DocPres (rule-based)									
GPT-4o$_{LM}$	–	–	76.42	13.28	–	2.98	2.33	3.24	2.85
Qwen2.5$_{LM}$	–	–	100.4	13.09	–	2.96	2.37	3.28	2.87
KCTV (template-based)									
GPT-4o$_{LM}$	–	80.0	68.48	10.27	–	2.49	2.94	3.57	3.00
Qwen2.5$_{LM}$	–	88.0	**41.41**	**16.76**	–	2.55	2.95	3.36	2.95
PPTAGENT *(ours)*									
GPT-4o$_{LM}$	GPT-4o$_{VM}$	**97.8**	721.54	10.17	7.48	3.25	3.24	4.39	3.62
Qwen2-VL$_{LM}$	Qwen2-VL$_{VM}$	43.0	265.08	13.03	7.32	3.13	**3.34**	4.07	3.51
Qwen2.5$_{LM}$	Qwen2-VL$_{VM}$	95.0	496.62	14.25	**6.20**	**3.28**	3.27	**4.48**	**3.67**

Table 5: Performance comparison of presentation generation methods, including DocPres, KCTV, and our proposed PPTAGENT. The best/second-best scores are **bolded**/underlined. Results are reported using existing metrics, including Success Rate (SR), Perplexity (PPL), Rouge-L, Fréchet Inception Distance (FID), and PPTEVAL.

• **Perplexity (PPL)** measures the likelihood of the model generating the given sequence. Using Llama-3-8B (Dubey et al., 2024), we calculate the average perplexity across all slides in a presentation. Lower perplexity scores indicate higher textual fluency (Bandyopadhyay et al., 2024).

• **Rouge-L (Lin, 2004)** evaluates textual similarity by measuring the longest common subsequence. While ROUGE typically needs golden labels, we use original documents as reference to test its effectiveness in their absence. We report the F1 score, and subsequently analyze its effectiveness.

• **FID (Heusel et al., 2017)** measures the visual similarity between the generated presentation and the reference presentation in the feature space.

• **PPTEVAL** employs GPT-4o to evaluate presentation quality across three dimensions: content, design, and coherence. We compute content and design scores by averaging across slides, while coherence is assessed at the presentation level.

4.5 Improvement by PPTAGENT (RQ1)

PPTAGENT Significantly Improves Overall Presentation Quality. PPTAGENT demonstrates statistically significant performance improvements over baseline methods across all three dimensions of PPTEVAL. Compared to the rule-based baseline (DocPres), PPTAGENT exhibits substantial improvements in both design and content dimensions (3.34 vs. 2.37, +40.9%; 3.28 vs. 2.98, +10.1%), as presentations generated by the DocPres method show minimal visual appeal. In comparison with the template-based baseline (KCTV), PPTAGENT also achieves notable improvements in both design and content (3.34 vs. 2.95, +13.2%; 3.28 vs. 2.55, +28.6%), underscoring the efficacy of the edit-based paradigm. Most notably, PPTAGENT shows a significant enhancement in the coherence dimension (4.48 vs. 3.57, +25.5% for DocPres; 4.48 vs. 3.28, +36.6% for KCTV). This improvement can be attributed to PPTAGENT 's comprehensive analysis of the structural role of slides.

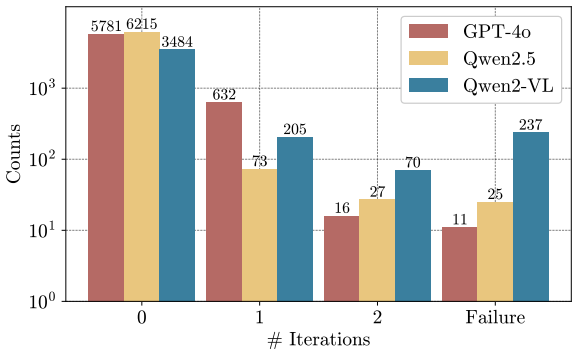

Figure 4: The number of iterative self-corrections required to generate a single slide under different models.

PPTAGENT Exhibits Robust Generation Performance. Our approach empowers LLMs to produce well-rounded presentations with a remarkable success rate, achieving $\geq 95\%$ success rate for both Qwen2.5$_{LM}$ and GPT-4o$_{LM}$, which is a significant improvement compared to KCTV (97.8% vs. 88.0%). Moreover, detailed performance analysis of PPTAGENT is illustrated in Appendix B, underscoring the versatility and robustness of our approach.

Self-Correction Proves Helpful Figure 4 shows the number of iterations required to generate a slide using different LLMs. Although GPT-4o exhibits superior self-correction capabilities, Qwen2.5 en-

Round	SYNTAX	INDEX	INST	HALLU	FUNCTION
0	14	1166	70	34	12
1	8	347	10	20	3
2	7	244	8	12	2

Table 6: Distribution of error types in PPTAGENT's slide generation process across self-correction rounds.

Setting	SR(%)	Content	Design	Coherence	Avg.
Ablation Studies					
PPTAGENT	95.0	3.28	3.27	**4.48**	3.67
w/o Outline	91.0	3.24	3.30	3.36	3.30
w/o Schema	78.8	3.08	3.23	4.04	3.45
w/o Structure	92.2	3.28	3.25	3.45	3.32
w/o CodeRender	74.6	3.27	3.34	4.38	3.66
Controlled Experiments					
Controlled	**100.0**	3.21	3.60	4.27	3.69
KCTV	88.0	2.55	2.95	3.36	2.95
Human	-	**4.01**	**3.95**	4.28	**4.08**

Table 7: Ablation studies and controlled experiments for PPTAGENT with the Qwen2.5$_{LM}$+Qwen2-VL$_{VM}$ configuration, illustrating the impact of each component. "Controlled" denotes PPTAGENT evaluated with the same template as KCTV, but without human annotations. "Human" represents the performance of presentations authored by humans.

counters fewer errors in the first generation. Moreover, we observed that Qwen2-VL experiences errors more frequently and struggles to correct them, likely due to its degraded language proficiency. Ultimately, all three models successfully corrected more than half of the errors, demonstrating that our iterative self-correction mechanism effectively enhances the robustness of the generation process.

Table 6 further illustrates how different error types evolve throughout the self-correction process. Our analysis reveals five distinct error categories: (1) SYNTAX errors arising from invalid code generation, (2) INDEX errors occurring when referencing non-existent slide elements, (3) INST errors resulting from violations of instruction constraints, (4) HALLU errors caused by references to not provided images or tables, and (5) FUNCTION errors stemming from incorrect API usage. Initially, INDEX errors constitute the vast majority (89.9%) of all errors, with 1,166 occurrences. Encouragingly, after two correction rounds, this number dramatically decreases to 244. The self-correction mechanism shows varied effectiveness, with INST and FUNCTION errors reducing by over 80%, while SYNTAX and HALLU errors persist more, decreasing by only 50% and 64.7% respectively. Overall, the total error count experiences a substantial decline from 1,296 to 273, validating the efficacy of our self-correction approach.

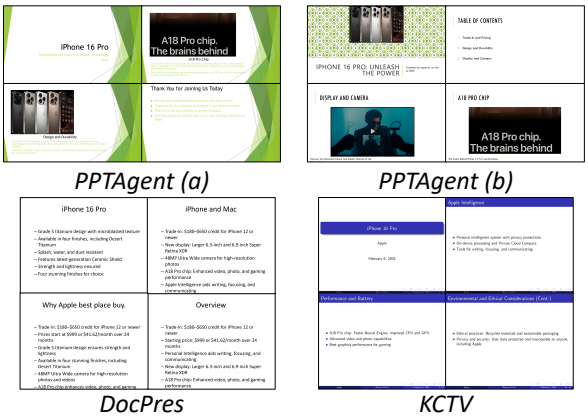

Figure 5: Comparative analysis of presentation generation across different methods: DocPres, KCTV, and PPTAGENT. *PPTAgent (a)* and *PPTAgent (b)* illustrate outputs generated by PPTAGENT using different reference presentations, demonstrating visually compelling outputs with stylistic diversity.

Ablation Study We conducted ablation studies on four settings: (1) randomly selecting a slide as the reference (w/o Outline), (2) omitting structural slides during outline generation (w/o Structure), (3) replacing the slide representation with the method proposed by Guo et al. (2023) (w/o CodeRender), and (4) removing guidance from content schemas (w/o Schema). All experiments were conducted using the Qwen2.5$_{LM}$+Qwen2-VL$_{VM}$ configuration.

As demonstrated in Table 7, our experiments reveal two key findings: 1) **The HTML-based representation significantly reduces interaction complexity**, evidenced by the substantial decrease in success rate from 95.0% to 74.6% when removing the Code Render component. 2) **The presentation analysis is crucial for generation quality**, as removing the outline and structural slides significantly degrades coherence (from 4.48 to 3.36/3.45) and eliminating the slide schema reduces the success rate from 95.0% to 78.8%.

Controlled Experiments Given that PPTAGENT operates with different input conditions with baselines, we conduct controlled experiments to minimize perturbations that could arise from diverse reference presentations, ensuring a fair comparison. In our controlled setup, PPTAGENT was supplied with the identical template[4] used by KCTV, but without any human annotations that KCTV requires. Furthermore, we also evaluated 50 human-

[4]The template is available at BeamerStyleSlides.

Correlation	Content	Design	Coherence	Avg.
Reference Analysis				
Pearson	0.43	0.77	0.11	0.44
Spearman	0.41	0.78	0.11	0.43
Agreement Analysis				
Pearson	0.70	0.90	0.55	0.71
Spearman	0.73	0.88	0.57	0.74

Table 8: Correlation scores across Content, Design, and Coherence dimensions, comparing (1) generated presentations with their reference counterparts, and (2) human ratings with LLM ratings. All presented data of similarity exhibit a p-value below 0.05, indicating a statistically significant level of confidence.

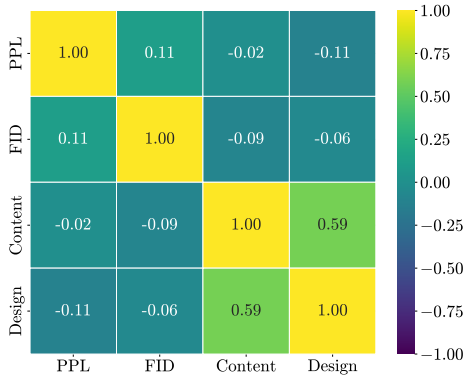

Figure 6: Correlation heatmap between existing automated evaluation metrics along with the content and design dimension in PPTEVAL.

authored presentations, which serve here as a reference benchmark for human performance.

As shown in Table 7, PPTAGENT outperforms KCTV under these controlled input conditions, highlighting the superiority of our proposed method. Moreover, its performance is comparable to human-authored presentations, indicating PPTAGENT's ability to generate practical outputs, although the limited capability of the current retrieval mechanism may constrain content richness.

4.6 Impact of Reference Presentations (RQ2)

Case Study We present representative examples of presentations generated under different methods in Figure 5. PPTAGENT generates presentations with superior quality on multiple dimensions. First, it effectively incorporates visual elements with contextually appropriate image placements, while maintaining concise and well-structured slide content. Second, it exhibits diversity in generating visually engaging slides under diverse references. In contrast, baseline methods (DocPres and KCTV) produce predominantly text-heavy slides with limited visual variation, constrained by their rule-based or template-based paradigms.

Correlational Findings PPTAGENT is designed to leverage reference presentations, particularly for aspects of style and layout. Table 8 presents the Pearson and Spearman correlations between the PPTEVAL scores of generated presentations with their reference counterparts. Notably, PPTAGENT effectively transfers design excellence from exemplars, leading to visually compelling outputs, as evidenced by a high Pearson correlation for the Design dimension. In contrast, the moderate and weak positive correlations for the Content and Coherence dimensions suggest that PPTAGENT's own capabilities in content generation and outline planning are more influential for these crucial elements.

4.7 Effectiveness of Evaluation Metrics (RQ3)

PPTEVAL Aligned with Human Preferences Although Chen et al. (2024a) and Kwan et al. (2024) have highlighted LLMs' impressive human-like discernment, validating their evaluations with human judgments for presentations remains crucial, especially since findings by Laskar et al. (2024) indicate LLMs may be inadequate for complex task evaluation. Table 8 shows the correlation of ratings between humans and LLMs. The average Pearson correlation of 0.71 exceeds the scores of other evaluation methods (Kwan et al., 2024), indicating that PPTEVAL aligns well with human preferences.

PPTEVAL Advances Traditional Metrics Existing metrics like ROUGE-L and FID, which evaluate through textual overlap or language fluency, often fail to reliably assess presentation quality, as they struggle with the fragmented and diverse slide content. Similarly, FID, which measures visual similarity to references, may not capture nuanced design appeal beyond mere conformity. Our analysis substantiates these limitations: Figure 6 shows only weak Pearson correlations between these traditional metrics and PPTEVAL's Content and Design dimensions. Furthermore, Table 5 reveals notable performance inconsistencies. For instance, KCTV scores best on ROUGE-L (16.76) and PPL (41.41), yet it receives a low PPTEVAL content score (2.55). Conversely, our method, with a ROUGE-L of 14.25 and PPL of 496.62, demonstrates substantially better content quality (3.28) as per PPTEVAL. Such discrepancies underscore the value of PPTEVAL,

for its dual capability of reliable label-free assessment and holistic evaluation of presentation coherence.

5 Related Works

Automated Presentation Generation Recent proposed methods for slide generation can be categorized into rule-based and template-based methods, depending on how they handle element placement and styling. Rule-based methods, such as those proposed by Mondal et al. (2024) and Bandyopadhyay et al. (2024), often focus on enhancing textual content but neglect the visual-centric nature of presentations, leading to outputs that lack engagement. Template-based methods, including Cachola et al. (2024) and industrial solutions like Tongyi, rely on predefined templates to create visually appealing presentations. However, their dependence on extensive manual effort for template annotation significantly limits scalability and flexibility.

LLM Agent Numerous studies (Deng et al., 2024; Li et al., 2024; Tang et al., 2025) have explored the potential of LLM to act as agents assisting humans in a wide range of tasks. For example, Wang et al. (2024b) demonstrate the capability of LLMs accomplish tasks by generating executable actions. Furthermore, Guo et al. (2023) demonstrated the potential of LLMs in automating presentation-related tasks through API integration.

LLM as a Judge LLMs have exhibited strong capabilities in instruction following and context perception, which has led to their widespread adoption as judges (Chen et al., 2025; Liu et al., 2023; Zheng et al., 2023). Chen et al. (2024a) demonstrated the feasibility of using MLLMs as judges, while Kwan et al. (2024) proposed a multi-dimensional evaluation framework. Additionally, Ge et al. (2025) investigated assessing single-slide quality, but their work lacks evaluating presentation from a holistic perspective.

6 Conclusion

In this paper, we introduce PPTAGENT, which formulates presentation generation as a two-stage presentation editing task completed through LLMs' abilities to understand and generate code. Moreover, we propose PPTEVAL to comprehensively evaluate the quality of the presentation. Our experiments across data from multiple domains have demonstrated the superiority of our method. This research provides a new paradigm for generating slides under unsupervised conditions and offers insights for future work in presentation generation.

Limitations

While PPTAGENT demonstrates promising capabilities in presentation generation, several limitations remain. First, despite achieving a high success rate (>95%) on our dataset, the model occasionally fails to generate presentations, which could limit its practicality. Second, the limited capability of the current retrieval system may restrict content richness in the generated outputs. Third, although PPTAGENT effectively leverages reference presentations to improve the visual appeal, it does not fully utilize visual information for fine-grained slide design refinement, such as optimizing element placement to prevent overlapping. Future work can focus on enhancing the generation robustness, incorporating a more capable retrieval mechanism, and investigating methods for advanced slide design refinement using visual cues.

Ethical Considerations

In the construction of *Zenodo10K*, we utilized the publicly available API to scrape data while strictly adhering to the licensing terms associated with each artifact. Specifically, artifacts that were not permitted for modification or commercial use under their respective licenses were filtered out to ensure compliance with intellectual property rights. Additionally, all annotation personnel involved in the project were compensated at rates exceeding the minimum wage in their respective cities, reflecting our commitment to fair labor practices and ethical standards.

Acknowledgments

We sincerely thank the reviewers for their insightful comments and valuable suggestions. This work was supported by Beijing Natural Science Foundation (L243006), the Natural Science Foundation of China (No. 62476265, 62306303), the Basic Research Program of ISCAS (Grant No. ISCAS-ZD-202401).

References

Sambaran Bandyopadhyay, Himanshu Maheshwari, Anandhavelu Natarajan, and Apoorv Saxena. 2024.

Enhancing presentation slide generation by llms with a multi-staged end-to-end approach. *arXiv preprint arXiv:2406.06556*.

Andrea Barrick, Dana Davis, and Dana Winkler. 2018. Image versus text in powerpoint lectures: Who does it benefit? *Journal of Baccalaureate Social Work*, 23(1):91–109.

Isabel Alyssa Cachola, Silviu Cucerzan, Allen Herring, Vuksan Mijovic, Erik Oveson, and Sujay Kumar Jauhar. 2024. Knowledge-centric templatic views of documents. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 15460–15476, Miami, Florida, USA. Association for Computational Linguistics.

Dongping Chen, Ruoxi Chen, Shilin Zhang, Yinuo Liu, Yaochen Wang, Huichi Zhou, Qihui Zhang, Pan Zhou, Yao Wan, and Lichao Sun. 2024a. Mllm-as-a-judge: Assessing multimodal llm-as-a-judge with vision-language benchmark. *arXiv preprint arXiv:2402.04788*.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024b. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *arXiv preprint arXiv:2402.03216*.

Jiawei Chen, Xinyan Guan, Qianhao Yuan, Guozhao Mo, Weixiang Zhou, Yaojie Lu, Hongyu Lin, Ben He, Le Sun, and Xianpei Han. 2025. Consistentchat: Building skeleton-guided consistent dialogues for large language models from scratch. *arXiv preprint arXiv:2506.03558*.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Sam Stevens, Boshi Wang, Huan Sun, and Yu Su. 2024. Mind2web: Towards a generalist agent for the web. *Advances in Neural Information Processing Systems*, 36.

Nancy Duarte. 2008. *Slide: ology: The art and science of creating great presentations*, volume 1. O'Reilly Media Sebastapol.

Nancy Duarte. 2010. *Resonate: Present visual stories that transform audiences*. John Wiley & Sons.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

European Organization For Nuclear Research and OpenAIRE. 2013. Zenodo.

Weixi Feng, Wanrong Zhu, Tsu-jui Fu, Varun Jampani, Arjun Akula, Xuehai He, Sugato Basu, Xin Eric Wang, and William Yang Wang. 2024. Layoutgpt: Compositional visual planning and generation with large language models. *Advances in Neural Information Processing Systems*, 36.

Tsu-Jui Fu, William Yang Wang, Daniel McDuff, and Yale Song. 2022. Doc2ppt: Automatic presentation slides generation from scientific documents. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(1):634–642.

Jiaxin Ge, Zora Zhiruo Wang, Xuhui Zhou, Yi-Hao Peng, Sanjay Subramanian, Qinyue Tan, Maarten Sap, Alane Suhr, Daniel Fried, Graham Neubig, et al. 2025. Autopresent: Designing structured visuals from scratch. *arXiv preprint arXiv:2501.00912*.

Michael Robert Gryk. 2022. Human readability of data files. *Balisage series on markup technologies*, 27.

Xinyan Guan, Yanjiang Liu, Hongyu Lin, Yaojie Lu, Ben He, Xianpei Han, and Le Sun. 2024. Mitigating large language model hallucinations via autonomous knowledge graph-based retrofitting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18126–18134.

Yiduo Guo, Zekai Zhang, Yaobo Liang, Dongyan Zhao, and Duan Nan. 2023. Pptc benchmark: Evaluating large language models for powerpoint task completion. *arXiv preprint arXiv:2311.01767*.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. 2017. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30.

Ryo Kamoi, Yusen Zhang, Nan Zhang, Jiawei Han, and Rui Zhang. 2024. When can llms actually correct their own mistakes? a critical survey of self-correction of llms. *Transactions of the Association for Computational Linguistics*, 12:1417–1440.

Wai-Chung Kwan, Xingshan Zeng, Yuxin Jiang, Yufei Wang, Liangyou Li, Lifeng Shang, Xin Jiang, Qun Liu, and Kam-Fai Wong. 2024. Mt-eval: A multi-turn capabilities evaluation benchmark for large language models. *Preprint*, arXiv:2401.16745.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pages 611–626.

Md Tahmid Rahman Laskar, Sawsan Alqahtani, M Saiful Bari, Mizanur Rahman, Mohammad Abdullah Matin Khan, Haidar Khan, Israt Jahan, Amran Bhuiyan, Chee Wei Tan, Md Rizwan Parvez, Enamul Hoque, Shafiq Joty, and Jimmy Huang. 2024. A systematic survey and critical review on evaluating large language models: Challenges, limitations, and recommendations. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 13785–13816, Miami, Florida, USA. Association for Computational Linguistics.

Yanda Li, Chi Zhang, Wanqi Yang, Bin Fu, Pei Cheng, Xin Chen, Ling Chen, and Yunchao Wei. 2024. Appagent v2: Advanced agent for flexible mobile interactions. *arXiv preprint arXiv:2408.11824*.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-eval: NLG evaluation using gpt-4 with better human alignment. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2511–2522, Singapore. Association for Computational Linguistics.

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

Ishani Mondal, S Shwetha, Anandhavelu Natarajan, Aparna Garimella, Sambaran Bandyopadhyay, and Jordan Boyd-Graber. 2024. Presentations by the humans and for the humans: Harnessing llms for generating persona-aware slides from documents. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2664–2684.

Athar Sefid, Prasenjit Mitra, and Lee Giles. 2021. Slidegen: an abstractive section-based slide generator for scholarly documents. In *Proceedings of the 21st ACM Symposium on Document Engineering*, pages 1–4.

Edward Sun, Yufang Hou, Dakuo Wang, Yunfeng Zhang, and Nancy XR Wang. 2021. D2s: Document-to-slide generation via query-based text summarization. *arXiv preprint arXiv:2105.03664*.

Hao Tang, Darren Key, and Kevin Ellis. 2025. Worldcoder, a model-based llm agent: Building world models by writing code and interacting with the environment. *Advances in Neural Information Processing Systems*, 37:70148–70212.

VikParuchuri. 2023. marker.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. 2024a. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*.

Xingyao Wang, Yangyi Chen, Lifan Yuan, Yizhe Zhang, Yunzhu Li, Hao Peng, and Heng Ji. 2024b. Executable code actions elicit better llm agents. *arXiv preprint arXiv:2402.01030*.

Bichen Wu, Chenfeng Xu, Xiaoliang Dai, Alvin Wan, Peizhao Zhang, Zhicheng Yan, Masayoshi Tomizuka, Joseph Gonzalez, Kurt Keutzer, and Peter Vajda. 2020. Visual transformers: Token-based image representation and processing for computer vision. *Preprint*, arXiv:2006.03677.

Tong Wu, Guandao Yang, Zhibing Li, Kai Zhang, Ziwei Liu, Leonidas Guibas, Dahua Lin, and Gordon Wetzstein. 2024. Gpt-4v (ision) is a human-aligned evaluator for text-to-3d generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22227–22238.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. 2024. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36:46595–46623.

A Data Preprocessing

To maintain a reasonable cost, we selected presentations ranging from 12 to 64 pages and documents with text lengths from 2,048 to 20,480 characters. We extracted both textual and visual content from the source documents using VikParuchuri (2023). The extracted text was then organized into sections. For visual content, we utilize Qwen2-VL to generate image captions, which then guide the image selection process. To minimize redundancy, we removed duplicate images whose embeddings exhibited cosine similarity above 0.85. At the slide level, we further deduplicated by removing slides if their text embedding similarity to the preceding slide was over 0.8, as suggested by Fu et al. (2022).

B Detailed Performance of PPTAGENT

B.1 Domain Versatility

We present a detailed performance analysis of the $\text{Qwen2.5}_{\text{LM}}$+$\text{Qwen2-VL}_{\text{VM}}$ configuration across various domains in Table 9, which underscores the versatility and robustness of our approach.

Domain	SR (%)	PPL	FID	PPTEval
Culture	93.0	185.3	5.00	3.70
Education	94.0	249.0	7.90	3.69
Science	96.0	500.6	6.07	3.56
Society	95.0	396.8	5.32	3.59
Tech	97.0	238.7	6.72	3.74

Table 9: Evaluation results under the configuration of $\text{Qwen2-VL}_{\text{LM}}$+$\text{Qwen2-VL}_{\text{VM}}$ in different domains, using the success rate (SR), PPL, FID and the average PPTEval score across three evaluation dimensions.

B.2 Weighted Performance

To account for variations in generation success rates, we also analyze success rate-weighted performance scores, where failed generations receive a PPTEVAL score of 0. This approach, with results detailed in Table 10 and 11, highlights how lower success rates can significantly impact the overall effectiveness of methods.

When considering these weighted metrics, GPT-4o consistently demonstrates outstanding performance across various evaluation criteria, showcasing its advanced capabilities. Although both serve as multimodal models, Qwen2-VL's linguistic proficiency can be compromised by its multimodal post-training, whereas GPT-4o maintained strength in language-centric tasks. In particular, the introduction of Qwen2.5 substantially mitigates such linguistic deficiencies. The weighted performance of this combination is on par with that of GPT-4o, achieving the overall best performance among the tested configurations. This underscores the significant potential of highly capable open-source LLMs, like Qwen2.5, as competitive agents for complex tasks such as presentation generation.

Configuration		PPTEval			
Language Model	Vision Model	Content↑	Design↑	Coherence↑	Avg.↑
DocPres (rule-based)					
$\text{GPT-4o}_{\text{LM}}$	–	2.98	2.33	3.24	2.85
$\text{Qwen2.5}_{\text{LM}}$	–	2.96	2.37	3.28	2.87
KCTV (template-based)					
$\text{GPT-4o}_{\text{LM}}$	–	1.99	2.35	2.85	2.40
$\text{Qwen2.5}_{\text{LM}}$	–	2.24	2.59	2.95	2.59
PPTAGENT *(ours)*					
$\text{GPT-4o}_{\text{LM}}$	$\text{GPT-4o}_{\text{VM}}$	**3.17**	**3.16**	4.20	**3.54**
$\text{Qwen2-VL}_{\text{LM}}$	$\text{Qwen2-VL}_{\text{VM}}$	1.34	1.43	1.75	1.50
$\text{Qwen2.5}_{\text{LM}}$	$\text{Qwen2-VL}_{\text{VM}}$	3.11	3.10	**4.25**	3.48

Table 10: Weighted Performance comparison of presentation generation methods, including DocPres, KCTV, and our proposed PPTAGENT. Results are evaluated using Success Rate (SR), Perplexity (PPL), Rouge-L, Fr'echet Inception Distance (FID), and SR-weighted PPTEval.

Setting	SR(%)	Content	Design	Coherence	Avg.
PPTAGENT	**95.0**	3.11	3.10	4.25	3.48
w/o Outline	91.0	2.94	3.00	3.05	3.00
w/o Schema	78.8	2.42	2.54	3.18	2.71
w/o Structure	92.2	**3.02**	2.99	3.18	3.06
w/o CodeRender	74.6	2.43	2.49	3.26	2.73

Table 11: Ablation analysis of PPTAGENT utilizing the $\text{Qwen2.5}_{\text{LM}}$+$\text{Qwen2-VL}_{\text{VM}}$ configuration, with PPTEval scores weighted by success rate to demonstrate each component's contribution.

B.3 Score Distribution

We further investigated the score distribution of generated presentations to compare the performance characteristics across different methods, as shown in Figure 7. Constrained by their rule-based or template-based paradigms, baseline methods typically exhibit limited diversity in both content and design scores, with these scores predominantly concentrated at lower levels (e.g., 2 and 3). In contrast, PPTAGENT demonstrates a more favorable and dispersed score distribution, with a significant majority of its presentations (>80%) achieving scores of 3 or higher in these dimensions. Furthermore, PPTAGENT's dedicated handling and strategic use of structural slides contribute to its notably supe-

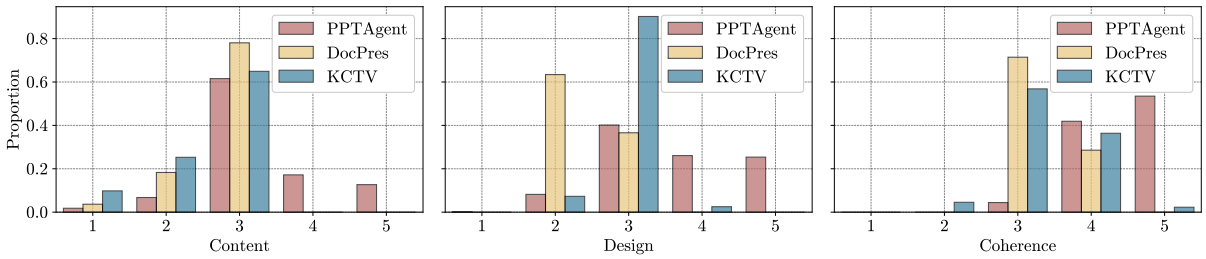

Figure 7: Score distributions of presentations generated by PPTAGENT, DocPres, and KCTV across the three evaluation dimensions: Content, Design, and Coherence, as assessed by PPTEVAL.

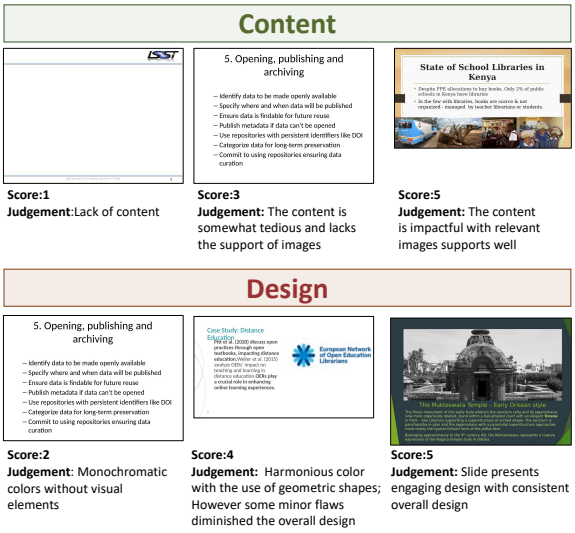

Figure 8: Scoring Examples of PPTEVAL.

rior coherence scores; over 80% of its presentations achieve coherence scores of 4 or above.

C Details of PPTEVAL

We recruited four graduate students through a Shanghai-based crowdsourcing platform to evaluate a total of 250 presentations: 50 randomly selected from *Zenodo10K* representing real-world presentations, along with two sets of 100 presentations generated by the baseline method and our approach, respectively. Following the evaluation framework proposed by PPTEVAL, assessments were conducted across three dimensions using the scoring criteria detailed in Appendix F. Evaluators were provided with converted slide images, scored them individually, and then discussed the results to reach a consensus on the final scores.

Moreover, we measured inter-rater agreement using Fleiss' Kappa, with an average score of 0.59 across three dimensions (0.61, 0.61, 0.54 for Content, Design, and Coherence, respectively), indicating satisfactory agreement (Kwan et al., 2024) among evaluators. Representative scoring examples are shown in Figure 8.

We provided a detailed illustration as below:

Content: The content dimension evaluates the information presented on the slides, focusing on both text and images. We assess content quality from three perspectives: the amount of information, the clarity and quality of textual content, and the support provided by visual content. High-quality textual content is characterized by clear, impactful text that conveys the proper amount of information. Additionally, images should complement and reinforce the textual content, making the information more accessible and engaging. To evaluate content quality, we employ MLLMs on slide images, as slides cannot be easily comprehended in a plain text format.

Design: Good design not only captures attention but also enhances content delivery. We evaluate the design dimension based on three aspects: color schemes, visual elements, and overall design. Specifically, the color scheme of the slides should have a clear contrast to highlight the content while maintaining harmony. The use of visual elements, such as geometric shapes, can make the slide design more expressive. Finally, good design should adhere to basic design principles, such as avoiding overlapping elements and ensuring that the design does not interfere with content delivery.

Coherence: Coherence is essential for maintaining audience engagement in a presentation. We evaluate coherence based on the logical structure and the contextual information provided. Effective coherence is achieved when the model constructs a captivating storyline, enriched with contextual information that enables the audience to follow the content seamlessly. We assess coherence by analyzing the logical structure and contextual information extracted from the presentation.

D Slide Clustering

We present our hierarchical clustering algorithm for layout analysis in Algorithm 1, where slides are grouped into clusters using a similarity threshold θ of 0.65. To focus exclusively on layout patterns and minimize interference from specific content, we preprocess the slides by replacing text content with a placeholder character ("a") and substituting image elements with solid-color backgrounds. Then, we compute the similarity matrix using cosine similarity based on the ViT embeddings of converted slide images between each slide pair. Figure 9 illustrates representative examples from the resulting slide clusters.

Algorithm 1 Slides Clustering Algorithm

Input: Similarity matrix of slides $S \in \mathbb{R}^{N \times N}$, similarity threshold θ
Initialize: $C \leftarrow \emptyset$
while $\max(S) \geq \theta$ **do**
  $(i, j) \leftarrow \arg\max(S)$ ▷ Find the most similar slide pair
  if $\exists c_k \in C$ such that $(i \in c_k \vee j \in c_k)$ **then**
    $c_k \leftarrow c_k \cup \{i, j\}$ ▷ Merge into existing cluster
  else
    $c_{\text{new}} \leftarrow \{i, j\}$ ▷ Create new cluster
    $C \leftarrow C \cup \{c_{\text{new}}\}$
  end if
  Update S:
    $S[:, i] \leftarrow 0$, $S[i, :] \leftarrow 0$
    $S[:, j] \leftarrow 0$, $S[j, :] \leftarrow 0$
end while
Return: C

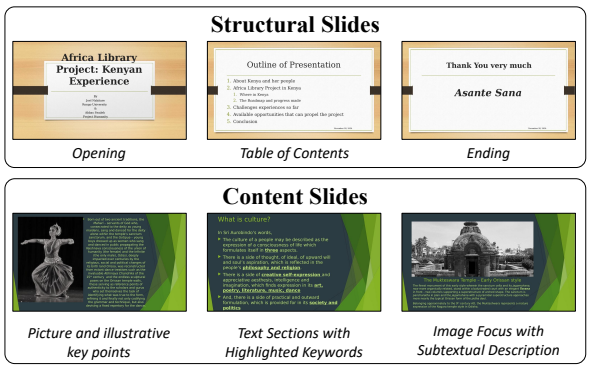

Figure 9: Example of slide clusters.

E Code Interaction

For visual reference, Figure 10 illustrates a slide rendered in HTML format, while Figure 11 displays its excerpt (first 60 lines) of the XML representation (out of 1,006 lines).

F Prompts

F.1 Prompts for Presentation Analysis

The prompts used for presentation analysis are illustrated in Figures 12, 13, and 14.

F.2 Prompts for Presentation Generation

The prompts used for generating presentations are shown in Figures 15, 16, and 17.

F.3 Prompts for PPTEVAL

The prompts used in PPTEVAL are shown in Figure 18, 19, 20, 21, 22 and 23.

Figure 10: Example of rendering a slide into HTML format.

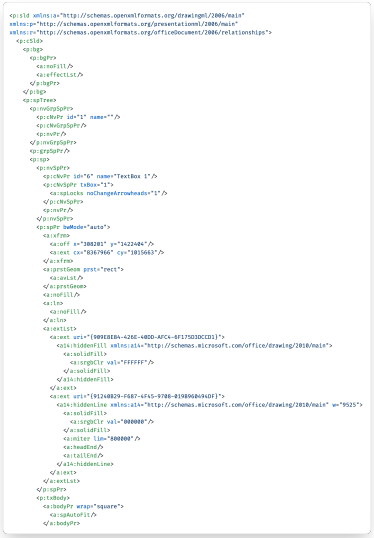

Figure 11: The first 60 lines of the XML representation of a presentation slide (out of 1,006 lines).

System Message:
You are an expert presentation analyst specializing in categorizing PowerPoint slides, particularly skilled at identifying structural slides (such as Opening, Transitions, and Ending slides) that guide the flow of the presentation. Please follow the specified output format strictly when categorizing the slides.

Prompt:
Objective: Analyze a set of slides provided in plain text format. Your task is to identify structural slides (such as Opening and Ending) based on their content and categorize all other slides under "Content."

Instructions:
1. Categorize structural slides in the presentation (such as Opening, Ending); assign all other slides to "Content."
2. Category names for structural slides should be simple, reflect their function, and contain no specific entity names.
3. Opening and Ending slides are typically located at the beginning or end of the presentation and may consist of only one slide.
4. Other transition categories must contain multiple slides with partially identical text.

Output format requirements:
Use the Functional key to group all categorized structural slides, with category names that reflect only the slide's function (e.g., "Opening," "Ending") and do not describe any specific content.
Use the Content key to list all slides that do not fall into structural categories.

Example output:
```json
{
  "functional": {
    "opening": [1],
    "table of contents": [2, 5],
    "section header": [3, 6],
    "ending": [10]
  },
  "content": [4, 7, 8, 9]
}
```

Ensure that all slides are included in the categorization, with their corresponding slide numbers listed in the output.

Input: {{slides}}

Output:

Figure 12: Illustration of the prompt used for clustering structural slides.

System Message:
You are a helpful assistant

Prompt:
Analyze the content layout and media types in the provided slide images.
Your objective is to create a concise, descriptive title that captures purely the presentation pattern and structural arrangement of content elements.
Requirements:
Focus on HOW content is structured and presented, not WHAT the content is
Describe the visual arrangement and interaction between different content types (text, images, diagrams, etc.)

Avoid:
Any reference to specific topics or subjects
Business or industry-specific terms
Actual content descriptions

You cannot use the following layout names:
{{ existed_layoutnames }}

Example Outputs:
Hierarchical Bullet Points with Central Image
Presentation of Evolution Through a Timeline
Analysis Displayed Using a Structured Table
Growth Overview Illustrated with Multiple Charts
Picture and illustrative key points
Layout
Output: Provide a one-line layout pattern title.

Figure 13: Illustration of the prompt used to infer layout patterns.

System Message:
You are a helpful assistant

Prompt:
Please analyze the slide elements and create a structured template schema in JSON format. The schema should:

1. Identify key content elements (both text and images) that make up the slide
2. For each element, specify:
  - "description": A clear description of the element's purpose, do not mention any detail
  - "type": "text" or "image" determined that according the tag of element: "image" is assigned for <img> tags
  - "data":
    * For text elements: The actual text content as string or array in paragraph level(<p> or <li>), merge inline text segments()
    * For image elements: Use the `alt` attribute of the <img> tag as the data of the image

Example format:
```
{
  "element_name": {
    "description": "purpose of this element", # do not mention any detail, just purpose
    "type": "text" or "image",
    "data": "actual text" or "<type>:<50-word description>" # detail here, cannot be empty or null
        or ["text1", "text2"]  # Multiple text elements
        or ["logo:...", "logo:..."]  # Multiple image elements
  }
}
Input:
{{slide}}
Please provide a schema that could be used as a template for creating similar slides.
```

Figure 14: Illustration of the prompt used to extract the slide schema.

System Message:
You are a professional presentation designer tasked with creating structured PowerPoint outlines. Each slide outline should include a slide title, a suitable layout from provided options, and concise explanatory notes. Your objective is to ensure that the outline adheres to the specified slide count and uses only the provided layouts. The final deliverable should be formatted as a JSON object. Please ensure that no layouts other than those provided are utilized in the outline.

Prompt:
Steps:

1. Understand the JSON Content:
  Carefully analyze the provided JSON input.
  Identify key sections and subsections.

{{ json_content }}

2. Generate the Outline:
  Ensure that the number of slides matches the specified requirement.
  Keep the flow between slides logical and ensure that the sequence of slides enhances understanding.
  Make sure that the transitions between sections are smooth through functional layouts.
  Carefully analyze the content and media types specified in the provided layouts.

For each slide, provide:
A Slide Title that clearly represents the content.
A Layout selected from provided layouts tailored to the slide's function.
Slide Description, which should contain concise and clear descriptions of the key points.

Please provide your output in JSON format.

Example Output:
```
{
  "Opening of the XX": {
    "layout": "layout1(media_type)",
    "subsection_keys": [],
    "description": "..."
  },
  "Introduction to the XX": {
    "layout": "layout2(media_type)", # select from given layouts(functional or content)
    "subsection_keys": ["Title of Subsection 1.1", "Title of Subsection 1.2"],
    "description": "..."
  }
}

Input:
Number of Slides: {{ num_slides }}
Image Information:
{{ image_information }}

# you can only use the following layouts
Content Layouts:
{{ layouts }}
Functional Layouts:
{{ functional_keys }}

Output:
```

Figure 15: Illustration of the prompt used for generating the outline.

System Message:
You are an Editor agent for presentation content. You transform reference text and available images into structured slide content following schemas. You excel at following schema rules like content length and ensuring all content is strictly derived from provided reference materials. You never generate new content or use images not explicitly provided.

Prompt:
Generate slide content based on the provided schema.
Each schema element specifies its purpose, and its default quantity.

Requirements:
1. Content Generation Rules:
  - Follow default_quantity for elements, adjust when necessary
  - All generated content must be based on reference text or image information
  - Ensure text content meets character limits
  - Generated text should use concise and impactful presentation style
  - For image elements, data should be the image path # eg: "images/logo.png"
  - Type of images should be a critical factor of image selection, if no relevant image(similar type or purpose) provided, leave it blank

2. Core Elements:
  - Must extract essential content from reference text (e.g., slide_title, main_content) and maintain semantic consistency
  - Must include images that support the main content (e.g., diagrams for explanations, visuals directly discussed in text)

3. Supporting Elements (e.g., presenters, logo images):
  - Generate only when relevant content exists in reference text or image information

Generate content for each element and output in the following format:
```
{
  "element1": {
    "data": ["text1", "text2"] for text elements
        or ["/path/to/image", "..."] for image elements
  },
}

Input:
Schema:
{{schema}}

Outline of Presentation:
{{outline}}

Metadata of Presentation:
{{metadata}}

Reference Text:
{{text}}

Available Images:
{{images_info}}

Output: the keys in generated content should be the same as the keys in schema
```

Figure 16: Illustration of the prompt used for generating slide content.

System Message:
You are a Code Generator agent specializing in slide content manipulation. You precisely translate content edit commands into API calls by following HTML structure, distinguishing between tags, and maintaining proper parent-child relationships to ensure accurate element targeting.

Prompt:
Generate the sequence of API calls based on the provided commands, ensuring compliance with the specified rules and precise execution.
You must determine the parent-child relationships of elements based on indentation and ensure that all and <img> elements are processed, leaving no unhandled content.

Each command follows this format: (element_class, type, quantity_change: int, old_data, new_data).

Steps

1. Quantity Adjustment:
- quantity_change Rules:
- If quantity_change = 0, do not perform clone_paragraph or del_span operations. Only replace the content.
- If quantity_change > 0, use clone_paragraph to add the corresponding number of paragraphs:
- When cloning, prioritize paragraphs from the same element_class that already have special styles (e.g., bold, color) if available.
- The paragraph_id for newly cloned paragraphs should be the current maximum paragraph_id of the parent element plus 1, while retaining the span_id within the cloned paragraph unchanged.
- If quantity_change < 0, use del_span or del_image to reduce the corresponding number of elements. Always ensure to remove span elements from the end of the paragraph first.
Restriction:
- Each command's API call can only use either clone_paragraph or del_span/del_image according to the `quantity_change`, but not both.
2. Content Replacement:
- Text Content: Use replace_span to sequentially distribute new content into one or more elements within a paragraph. Select appropriate tags for emphasized content (e.g., bold, special color, larger font).
- Image Content: Use replace_image to replace image resources.
3. Output Format:
- Add comments to each API call group, explaining the intent of the original command and the associated element_class.
- For cloning operations, annotate the paragraph_id of the newly created paragraphs.

Available APIs

{{api_docs}}

Example Input:

Please output only the API call sequence, one call per line, wrapped in ```python and ```, with comments for corresponding commands.

Figure 17: Illustration of the prompt used for generating editing actions.

System Message:
You are a help assistant

Prompt:
Please describe the input slide based on the following three dimensions:
1. The amount of information conveyed
Whether the slide conveys too lengthy or too little information, resulting in a large white space without colors or images.
2. Content Clarity and Language Quality
Check if there are any grammatical errors or unclear expressions of textual content.
3. Images and Relevance
Assess the use of visual aids such as images or icons, their presence, and how well they relate to the theme and content of the slides.

Provide an objective and concise description without comments, focusing exclusively on the dimensions outlined above.

Figure 18: Illustration of the prompt used to describe content in PPTEval.

System Message:
You are a help assistant

Prompt:
Please describe the input slide based on the following three dimensions:
1. Visual Consistency
Describe whether any style diminished the readability, like border overflow or blur, low contrast, or visual noise.
2. Color Scheme
Analyze the use of colors in the slide, identifying the colors used and determining whether the design is monochromatic (black and white) or colorful (gray counts in).
3. Use of Visual Elements
Describe whether the slide include supporting visual elements, such as icons, backgrounds, images, or geometric shapes (rectangles, circles, etc.).

Provide an objective and concise description without comments, focusing exclusively on the dimensions outlined above.

Figure 19: Illustration of the prompt used to describe style in PPTEval.

System Message:
You are an expert presentation content extractor responsible for analyzing and summarizing key elements and metadata of presentations. Your task is to extract and provide the following information:

Prompt:
Scoring Criteria (Five-point scale):
1. Slide Descriptions: Provide a concise summary of the content and key points covered on each slide.
2. Presentation Metadata: Identify explicit background information(which means it should be a single paragraph, not including in other paragraphs), such as the author, speaker, date, and other directly stated details, from the opening and closing slides.

Example Output:
```
{
  "slide_1": "This slide introduces the xx, xx.",
  "slide_2": "...",
  "background": {
    "speaker": "speaker x",
    "date": "date x"
  }
}
```

Input:
{{presentation}}

Output:.

Figure 20: Illustration of the prompt used to extract content in PPTEval.

System Message:
You are an unbiased presentation analysis judge responsible for evaluating the quality of slide content. Please carefully review the provided slide image, assessing its content, and provide your judgement in a JSON object containing the reason and score. Each score level requires that all evaluation criteria meet the standards of that level.

Prompt:
Scoring Criteria (Five-Point Scale):

1 Point (Poor):
The text on the slides contains significant grammatical errors or is poorly structured, making it difficult to understand.

2 Points (Below Average):
The slides lack a clear focus, the text is awkwardly phrased, and the overall organization is weak, making it hard to engage the audience.

3 Points (Average):
The slide content is clear and complete but lacks visual aids, resulting in insufficient overall appeal.

4 Points (Good):
The slide content is clear and well-developed, but the images have weak relevance to the theme, limiting the effectiveness of the presentation.

5 Points (Excellent):
The slides are well-developed with a clear focus, and the images and text effectively complement each other to convey the information successfully.

Example Output:
```
{
 "reason": "xx",
 "score": int
}
```
Input: {{descr}}
Let's think step by step and provide your judgment.

Figure 21: Illustration of the prompt used to evaluate content in PPTEval.

System Message:
You are an unbiased presentation analysis judge responsible for evaluating the visual appeal of slides. Please carefully review the provided description of the slide, assessing their aesthetics only, and provide your judgment in a JSON object containing the reason and score. Each score level requires that all evaluation criteria meet the standards of that level.

Prompt:
Scoring Criteria (Five-point scale):

1 Point (Poor):
There is a conflict between slide styles, making the content difficult to read.

2 Points (Fair):
The slide uses monotonous colors(black and white), ensuring readability while lacking visual appeal.

3 Points (Average):
The slide employs a basic color scheme; however, it lacks supplementary visual elements such as icons, backgrounds, images, or geometric shapes(like rectangles), making it look plain.

4 Points (Good):
The slide uses a harmonious color scheme and contains some visual elements(like icons, backgrounds, images, or geometric shapes); however, minor flaws may exist in the overall design.

5 Points (Excellent):
The style of the slide is harmonious and engaging, the use of supplementary visual elements like images and geometric shapes enhances the slide's overall visual appeal.

Example Output:
```
{
 "reason": "xx",
 "score": int
}
```

Input: {{descr}}
Let's think step by step and provide your judgment.

Figure 22: Illustration of the prompt used to evaluate style in PPTEval.

System Message:
You are an unbiased presentation analysis judge responsible for evaluating the coherence of the presentation. Please carefully review the provided summary of the presentation, assessing its logical flow and contextual information, each score level requires that all evaluation criteria meet the standards of that level.

Prompt:
Scoring Criteria (Five-Point Scale)

1 Point (Poor):
Terminology are inconsistent, or the logical structure is unclear, making it difficult for the audience to understand.

2 Points (Fair):
Terminology are consistent and the logical structure is generally reasonable, with minor issues in transitions.

3 Points (Average):
The logical structure is sound with fluent transitions; however, it lacks basic background information.

4 Points (Good):
The logical flow is reasonable and include basic background information (e.g., speaker or acknowledgments/conclusion).

5 Points (Excellent):
The narrative structure is engaging and meticulously organized with detailed and comprehensive background information included.

Example Output:
```
{
  "reason": "xx",
  "score": int
}
```

Input:
{{presentation}}

Let's think step by step and provide your judgment, focusing exclusively on the dimensions outlined above and strictly follow the criteria.

Figure 23: Illustration of the prompt used to evaluate coherence in PPTEval.