Textual-to-Visual Iterative Self-Verification for Slide Generation

Anonymous ACL submission

Abstract

Generating presentation slides is a time-consuming task that urgently requires automation. Due to their limited flexibility and lack of automated refinement mechanisms, existing autonomous LLM-based agents face constraints in real-world applicability. In this work, we decompose the task of generating missing presentation slides into two key components: **content generation** and **layout generation**, aligning with the typical process of creating academic slides. For content generation, we introduce a content generation approach that enhances coherence and relevance by incorporating context from surrounding slides and leveraging section retrieval strategies. For layout generation, we propose a **textual-to-visual self-verification process** using a **LLM-based Reviewer + Refiner workflow**, transforming complex textual layouts into intuitive visual formats. This modality transformation simplifies the task, enabling accurate and human-like review and refinement. Experiments show that our approach significantly outperforms baseline methods in terms of alignment, logical flow, visual appeal, and readability.

1 Introduction

Effectively summarizing and presenting research findings through academic presentation slides is an essential part of scientific communication, enabling researchers to highlight key contributions and engage audiences at conferences and seminars (Guo et al., 2024; Mondal et al., 2024). However, creating these slides is a time-consuming process that requires extracting core information from lengthy papers, organizing it coherently, and designing visually consistent layouts across multiple slides (Fu et al., 2021). With the rapid growth in the volume of research, the demand for automated solutions has increased significantly. Recent advances in large language models (LLMs) (OpenAI, 2023; Touvron et al., 2023; Templeton et al., 2024) have demonstrated remarkable capabilities in mimicking human behavior for complex tasks (Hong et al., 2023; Park et al., 2023; Yao et al., 2022b; **?**) beyond text generation (Yao et al., 2022b,a; Xi et al., 2024; Yang et al., 2024). Building on these strengths, LLM-based agents offer a promising opportunity to automate tasks like slide generation (Zheng et al., 2025), reducing manual effort while ensuring coherence and visual quality.

Despite its potential, generating high-quality academic presentation slides presents two major challenges: **how to assign reasonable and adaptive layouts for generated content** and **how to ensure layout quality and consistency**.

The first challenge lies in generating layout information that adapts to the unique visual structure for different textual contents. Some methods focus solely on textual content, neglecting structural aspects like positioning, spacing, and alignment, leading to impractical outputs (Sun et al., 2021; Bandyopadhyay et al., 2024). Existing template-based methods provide a quick and straightforward solution by populating predefined slots with generated content. However, they overlook the unique structural style of each presentation, often leading to rigid layouts that break the visual coherence.

The second challenge lies in achieving consistent textual-visual results, complicated by the inherent difficulty of representing slide layouts in structured textual formats. Unlike visual representations, where spatial relationships and element alignment are easy to interpret, textual formats lack this visual clarity (Xu et al., 2024; Hu et al., 2024). This makes it difficult for models to fully comprehend the spatial and structural aspects of slide design, leading to frequent errors such as text overflow, misalignment, and inconsistent spacing.

Furthermore, correcting these errors directly in the textual format is non-trivial. Without a visual reference, detecting overlapping elements or misalignments becomes challenging, particularly in slides with complex layouts.

A key component of our framework is a textual-to-visual iterative self-verification process to refine initial outputs. The initial slide layouts are generated in a textual format, which—while structured and machine-readable—often contains errors due to the complexity of representing slide information in a non-visual form. Additionally, reviewing and refining these layouts in their original format is challenging and unintuitive. To address this, we introduce a **modality transformation** (Li et al., 2025) that converts the textual format into a visualized form. This transformation significantly reduces the complexity of the task, making it easier for the LLM-based Reviewer + Refiner workflow to detect and correct issues such as alignment and text overflow in a human-like, intuitive manner. The reviewer provides feedback by analyzing the visual representation of the slide layout. The feedback is then passed to the refiner, who applies the suggested adjustments to the structured layout in textual format. This iterative refinement process ensures higher-quality final outputs with improved coherence and visual consistency.

Our key contributions are as follows.

1. An agentic framework for slide generation including content and layout generation approaches, ensuring thematic consistency and visual coherence.

2. A textual-to-visual iterative self-verification process with modality transformation, enabling intuitive and accurate refinement for slide layout.

3. Extensive analyses and systematic evaluation, demonstrating the significant effectiveness and practical potential of our framework for automated academic slide generation.

2 Related Work

In this section, we introduce the background of the LLM-based agent and existed studies on slides generations.

2.1 LLM-based Agent

LLMs have demonstrated impressive capabilities for complicated, interactive tasks (Yao et al., 2022b,a; Xi et al., 2024; Yang et al., 2024). LLM-based autonomous agents have achieved remarkable progress in a wide range of domains, including logic reasoning (Qi et al., 2024; Khattab et al., 2022), tool use (Qin et al., 2024), and social activities (Park et al., 2023). The current paradigm of agents relies on the language intelligence of LLMs. The mainstream work pattern encompasses environment perceiving, planning, reasoning, and executing, forming a workflow to dive and conquer intricate challenges.

Empowered by the recent progress of multi-modal pre-training, those agents can understand image, video, and audio channels (Wu et al., 2023; Liu et al., 2023). (i) Visual knowledge can largely facilitate reasoning and is integrated into Chain-of-Thoughts (Zhang et al., 2023; Xu et al., 2024). (ii) Multi-modal reasoning enables divergent thinking cross modalities and takes advantage of those different modalities. Sketchpad (Hu et al., 2024) allows LLMs to draw drafts to assist its planning and reasoning, i.e., to draw auxiliary lines for geometry problems. Visualization-of-Thought (Wu et al., 2024) generates visual rationales for spatial reasoning tasks like mazes. For each stage of complex multi-modal tasks, selecting an appropriate modality as the main modality for reasoning can leverage the natural characteristics of the modality and stimulate the potential of LLMs (Park et al., 2025).

2.2 Slide Generation

Previous studies have explored extractive methods and simplified this task as sentence selection, e.g., to calculate the importance score and extract top sentences (Wang et al., 2017). With the development of small language models (Lewis et al., 2020; Raffel et al., 2020), slide generation is unified as abstractive, query-based document summarization (Sun et al., 2021).

Despite their early success, the emergence of LLMs exhibits exceptional performance and stimulates the demands of intelligent slide generation. Slide generation poses intricate challenges for autonomous agents, as it requires document reading comprehension and precise tool use to generate layouts. Pioneer work focuses on modifying target elements, asking agents to execute a series of specific instructions (Guo et al., 2024). The agent needs to understand the status of the slide, navigate to the element, and generate precise API calls. Recent studies first plan the outlines and then generate each page. To further control the style of presentations, Mondal et al. (2024) introduce a reward model trained on human feedback to guide both topic generation and content extraction. Considering the visual quality of slides, Bandyopadhyay et al. (2024) employ a visual LM to insert images.

DOC2PPT (Fu et al., 2021) integrates an object placer to predict the position and size of each element by training small models. PPTAgent (Zheng et al., 2025) directly utilizes slide templates to fix the layout and then fill textboxes, ensuring visual harmony and aesthetic appeal.

3 Methodology

In this section, we propose an LLM-based agentic workflow to automate the generation of content and layout for academic paper slides.

3.1 Task Formulation

We first formally define our slide generation task. In this task, a presentation is represented as a collection of slide pages, where each page consists of multiple elements. Each element $e \in E$ is a tuple (c, l), where c denotes the content (e.g., text, images, tables) and l specifies the corresponding layout information (e.g., position, size, font style).

Our **overall task** is to generate the missing slide $\hat{S}_i$ given the paper D, the missing slide topic T, and the partially available slide set $S = \{S_1, S_2, \ldots, S_n\}$.

Input The input consists of: 1. A paper $D = \{d_1, d_2, \ldots, d_m\}$, where d_i denotes a section or paragraph in the paper. 2. A missing slide topic T, describing the main focus of the missing slide. 3. A partially available slide set $S = \{S_1, S_2, \ldots, S_n\}$, where some slides $\hat{S}_i$ are missing. 4. The preceding slide S_{prev} and the following slide S_{next} as contextual information.

Output The output is a structured textual file $\hat{S}_i$, which describes the missing slide, including both content c and layout information l for each element $e \in E$. Formally,

$$\hat{S}_i = \{e_j = (c_j, l_j) \mid j = 1, 2, \ldots, k\}$$

where k is the number of elements in the generated slide. The generated textual file can be directly converted into a PowerPoint slide.

3.2 Slide Generation Framework

The process of creating a presentation typically involves two key stages: (1) identifying the core content that needs to be presented on each slide, and (2) arranging this information into a visually coherent and consistent layout.

The goal of content generation is to generate c_j for each element e_j based on the paper D, the missing slide's title t, and contextual information from the surrounding slides S_{prev} and S_{next}:

$$c_j = \mathcal{G}_{\text{content}}(D, t, S_{prev}, S_{next})$$

Here, $\mathcal{G}_{\text{content}}$ represents the content generation process, ensuring that the generated content is accurate, concise, and contextually relevant.

The layout generation task determines the layout l_j for each element $e_j = (c_j, l_j)$ to maintain visual consistency and readability. The initial layout draft $l_j^{(0)}$ is generated using the content c_j and contextual information from the surrounding slides:

$$l_j^{(0)} = \mathcal{G}_{\text{layout\_draft}}(c_j, S_{prev}, S_{next})$$

To refine the initial layout, a textual-to-visual iterative self-verification process is applied. The layout at step k ($l_j^{(k)}$) is visualized as Image($l_j^{(k)}$), allowing the LLM-based Reviewer + Refiner workflow to provide feedback and corrections:

$$l_j^{(k+1)} = \mathcal{G}_{\text{refine}}\left(l_j^{(k)}, \text{Image}(l_j^{(k)})\right)$$

This iterative process continues until the layout reaches the desired quality and visual coherence.

3.2.1 Content Generation

Determining the key contents on a slide page involves understanding paper structures, extracting critical texts and figures, and ensuring overall coherence for a logical flow and consistent style.

Our content generation stage adopts a multi-step process with three sub-modules: Text Retriever, Figure Extractor, and Content Generator, consisting of a pipeline to identify relevant text segments, recommend figures and tables, and then decide the contents to present.

Text Retriever We build a text retriever to retrieve the most relevant sections of the paper. The paper is divided into section-level granularity, with each segment represented and indexed as a dense embedding. Given the topic of a slide, the retriever selects the most relevant segments by calculating the cosine similarity between the dense embeddings of the slide topic and the indexed sections.

Figure Extractor Beyond the retrieved text, figure extractor focuses on extracting relevant figures to provide visual elements for the slide content. This process identifies references to figures and tables within the text (e.g., “Figure 1”, “Table 2”) and extracts their captions from the paper.

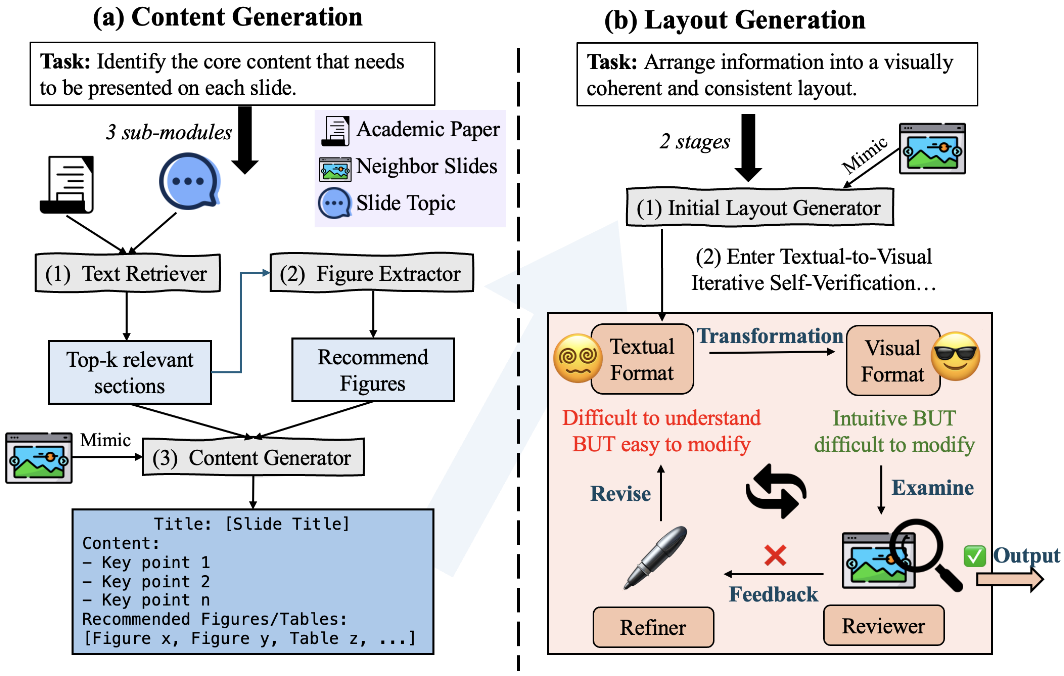

Figure 1: Overall Framework

Content Generator The LLM agent performs three sub-tasks based on the related text segments and recommended figures. First, it generates concise slide text aligned with the slide's topic and context. Second, it selects the most relevant figures and tables to complement the content and improve comprehension. Finally, it integrates surrounding slide content to maintain logical flow and ensure seamless transitions.

The results of the Content Generator above are aggregated for the following layout generation, where the focus shifts to organizing the content into a visually coherent and well-structured slide layout.

3.2.2 Layout Generation

Slide layouts need to be flexible and controllable, rather than fully randomized or constrained by rigid templates. However, generating adaptive layouts is challenging and prone to issues such as text overflow, misalignment, and inconsistent spacing, especially when handling diverse content and styles.

To address this, we design a **textual-to-visual iterative self-verification process**. The initial layout draft mimics surrounding slides for style consistency but remains difficult to review in its structured textual format. By converting the draft into a visual representation, i.e. an image. We design an LLM-based *Reviewer* + *Refiner* workflow that validates and refines the layout respectively, improving accuracy and coherence through iterative corrections.

Stage 1: Initial Layout Generation The initial attempt is conducted by directly asking the LLM to arrange the layout for each element of the generated contents, specifying each element's position, size, font, and color. We also append surrounding slide pages as demonstrations and carefully optimize the prompt to instruct the LLM to mimic their layout patterns for a visually consistent design. The layout is normalized as a JSON format.

While this initial layout serves as a foundation, our pilot experiments show that several factors contribute to potential errors:

(i) Textual slide layout is inherently complex, requiring detailed key-value pairs for positions, sizes, fonts, and colors. Any inconsistency in this structured data can cause significant visual defects.

(ii) LLMs lack direct visual feedback and cannot accurately assess how the generated layout will appear in its final form. Unlike models specifically trained for visual tasks, LLMs rely on textual context and structural patterns to predict layout information. This process is inherently limited, as it depends heavily on imitation and pattern recognition without understanding visual balance or spatial relationships. Consequently, the generated layouts may exhibit issues such as poor alignment, overlapping elements, or inconsistent spacing, which require further refinement to ensure high-quality results.

Stage 2: Textual-to-Visual Iterative Self-Verification To refine the initial layout, we introduce a self-verification process that combines modality transformation and a LLM-based agentic workflow.

Modality Transformation We first convert the initial textual output into a visualized slide. The initialized layout is written into a slide and saved as an image. To facilitate visual perception, each visualized element in the slide is enclosed in a colored bounding box with a unique **ID**, matching its corresponding element in the textual file. This visual augmentation simplifies the workload, largely relieving the burden of perception and enabling the Reviewer to quickly reference specific elements and detect potential issues.

Reviewer The Reviewer simulates how a human expert would evaluate slide quality, following a predefined set of evaluation criteria and adjustment rules. Specifically, it performs the following tasks: Object overlapping detection, Image quality and distortion analysis, Element bounding and text overflow correction, Element positioning and alignment, Text formatting consistency and Overall composition and visual balance

Each recommendation is output as a structured list of suggestions, identifying specific elements by their **ID** and providing precise numerical values for adjustments. For example, the Reviewer might suggest increasing a text box's height by 1.2x to accommodate overflowing text or shifting an image downward by 10% of its height to resolve an overlap. Such a definite, specific advice format makes it easier for the Refiner to implement precise corrections in the subsequent refinement stage.

Refiner The Refiner plays a role for execution, translating the Reviewer's visual feedback into precise modifications within the textual layout. To ensure accurate modifications, the Refiner follows a set of predefined rules based on the type of feedback received. For example, when the Reviewer suggests repositioning an element, the Refiner adjusts its bounding box coordinates accordingly while ensuring it remains within slide boundaries. Each rule is applied systematically based on the Reviewer's feedback. The Refiner's task is to modify only the necessary fields while maintaining the basic structure, resulting in a complete and refined file that reflects the intended adjustments.

Integration and Rendering The final output of this process is a refined JSON-formatted layout description that accurately represents the corrected slide. This JSON is passed to the rendering module to produce the final PowerPoint slide, ensuring that the layout visually reasonable and aligns with the overall presentation style.

4 Experiments

4.1 Dataset Construction

The dataset is sourced from the ACL 2024 In-Person Poster Session 1, with data collected from the public academic platform Underline. The dataset consists of academic papers and their corresponding PowerPoint slides in PDF format, covering various research topics in natural language processing. To facilitate processing and preserve format details, all data is uniformly converted into JSON format, containing element-level information such as text content, font styles, positions, and sizes. Text from papers was extracted using GROBID (Kermitt2, 2020). Figures and captions were extracted using PDFFigures 2.0 (Clark and Divvala, 2016).

4.2 Baseline

The baseline for Content Generation provides the full paper and the corresponding slide topic directly to the LLM, which generates content in a fixed format without retrieval or surrounding slide context. The baseline for Layout Generation generates the slide layout by directly using the generated content and the JSON layout information from surrounding slides. It does not mimic the style or structure of neighboring slides and lacks iterative refinement.

4.3 Implementation

We compare the performance of three large language models: **Llama-31-8B-Instruct** (Grattafiori et al., 2024), **GPT-4o** (OpenAI et al., 2024), and **Qwen-2.5-7B** (Qwen et al., 2025). The best-performing model is selected to generate the final structured content. In the layout generation module, both the Reviewer and Refiner modules are built on top of multimodal large language model.

For the retriever, we use the **Salesforce SFR-Embedding-Mistral** (Wang et al., 2024) retriever to compute similarity scores and select the top-k most relevant sections.

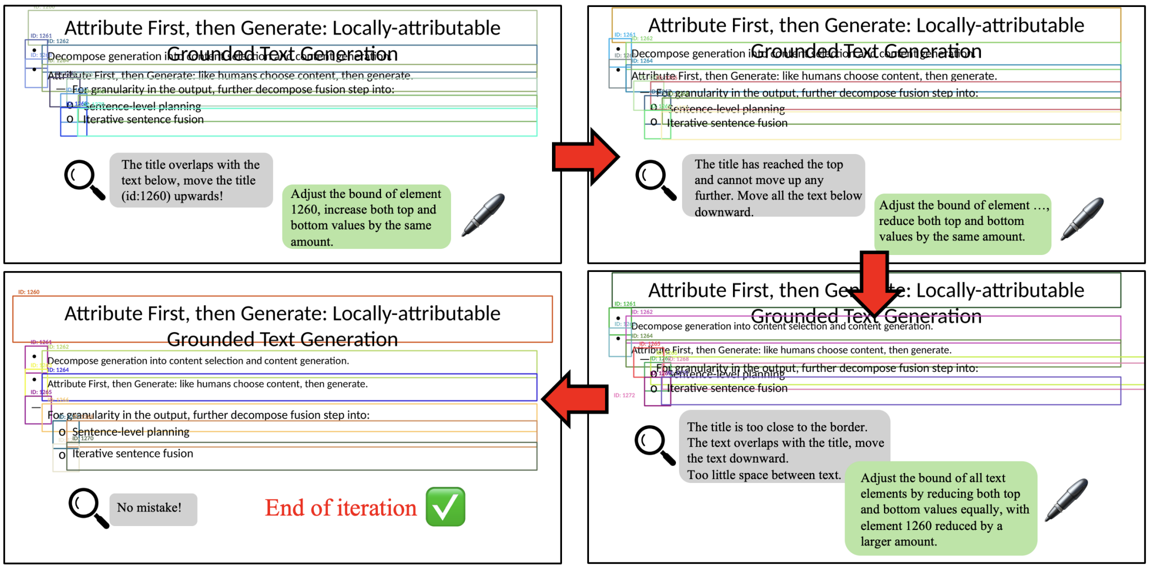

Figure 2: Iterative Layout Refinement in the Reviewer + Refiner Workflow

Our experiments are naturally organized in the form of ablations. In the **w/o Section Retriever** configuration, the model receives the entire paper as input without section-level retrieval. In the **w/o Neighbor Slides** configuration, the surrounding slide content is removed, which helps assess the role of contextual information in maintaining logical flow and consistency.

4.4 Evaluation

Our evaluation method measures both content generation and layout generation. The evaluation process combines quantitative metrics and structured qualitative assessment to ensure comprehensive analysis.

Content Evaluation We use ROUGE (Lin, 2004) as the primary evaluation metric to measure the similarity between the generated slide content and the author-provided reference slides.

Layout Evaluation We adopt LLM-as-Judge (Chen et al., 2024) to evaluate slide layouts across three levels:

◦ **Element Level**: Assesses alignment, spacing, and positioning of individual elements to ensure a well-structured layout.

◦ **Slide Level**: Focuses on logical flow and text-visual consistency, ensuring information is presented clearly and supported by relevant visuals.

◦ **Overall Impression**: Evaluates visual appeal and readability, ensuring cohesive design, appropriate font size, and clear charts for an accessible presentation.

4.5 Main Results

Content Generation Among the three models, GPT-4o demonstrates the most consistent and high performance, particularly in ROUGE-L F1 (21.97) and ROUGE-2 Recall (15.71). Although Llama-31-8B shows competitive performance in certain cases (e.g., ROUGE-1 Recall 47.74 for the Baseline), GPT-4o achieves a better balance between precision and recall. Qwen2.5-7B shows moderate performance, but its results are slightly more variable compared to the other models.

Layout Generation For layout evaluation, Table 2 summarizes the results of layout generation across three different configurations: Baseline, Textual-Based Refinement, and Our Method. The Reference Slide serves as a benchmark for assessing the quality of generated layouts.

Baseline: This configuration represents the initial layout generated by the model without any refinement. The layout is stored in a structured JSON format describing element positions, sizes, and other attributes. However, due to the complexity of multi-element layouts and the lack of direct visual feedback, this initial output often contains errors such as misalignment, text overflow, and inconsistent spacing.

Textual-Based Refinement: In this configuration, the initial JSON file is refined through an automated rule-based review. The Reviewer analyzes the JSON structure to detect layout issues, while the Refiner applies corrective actions directly to the JSON file. Although this approach improves some

LLM	Method	ROUGE-1			ROUGE-2			ROUGE-L		
		P	R	F1	P	R	F1	P	R	F1
Llama-31-8B	Baseline	24.56	47.74	28.02	8.94	19.96	10.34	17.54	37.58	20.46
	Proposed Method (3)	28.64	39.30	27.47	11.23	17.13	11.15	21.99	32.18	21.36
	Proposed Method (5)	28.52	42.63	28.40	11.38	19.33	11.68	21.76	34.99	21.97
	w/o Neighbor Slides	25.31	42.31	26.79	9.78	19.03	10.72	19.00	34.07	20.42
	w/o Section Retriever	30.06	42.04	29.35	12.44	19.45	12.54	23.19	34.85	22.99
GPT-4o	Baseline	23.29	43.97	25.65	7.15	16.86	8.20	16.23	34.09	18.31
	Proposed Method (3)	31.63	32.86	26.10	11.30	14.91	9.84	24.34	27.81	20.76
	Proposed Method (5)	31.75	37.68	28.39	10.89	15.71	10.28	24.09	30.60	21.97
	w/o Neighbor Pages	29.11	34.60	26.13	10.18	15.43	9.61	22.79	29.21	20.88
	w/o Section Retriever	32.48	37.68	28.36	11.15	15.88	10.05	24.45	30.35	21.64
Qwen2.5-7B	Baseline	24.27	44.92	26.02	9.06	19.69	10.10	17.89	36.24	19.65
	Proposed Method (3)	29.78	36.26	25.99	11.63	16.58	10.56	24.17	30.76	21.21
	Proposed Method (5)	28.31	37.17	26.01	10.29	15.71	9.87	21.60	30.21	20.18
	w/o Neighbor Pages	24.13	44.93	25.91	9.01	19.69	10.06	17.78	36.26	19.57
	w/o Section Retriever	31.47	36.77	27.92	12.60	17.11	11.60	24.66	30.39	22.14

Table 1: Evaluation results for content generation

metrics, such as **Coherence (3.4)**, it still struggles with **Visual Appeal (1.8)** and **Alignment (2.1)**, indicating the limitations of rule-based refinement without visual feedback.

Our Method: By introducing **modality transformation**, we convert the JSON layout into a fully visualized slide image, allowing the Reviewer + Refiner workflow to detect and correct issues more intuitively. This approach yields significant improvements, especially in **Alignment and Spacing (3.0)** and **Logical Flow (3.8)**, closely approaching the quality of the reference slides. Additionally, **Visual Appeal (2.8)** and **Readability (3.0)** show notable gains compared to the previous configurations.

The results indicate that incorporating the Reviewer + Refiner workflow and modality transformation significantly improves layout quality, especially in terms of visual appeal and overall readability.

5 Analysis

5.1 Ablation

Effect of Neighbor Slides Neighbor slides significantly impact the quality of content generation. For instance, removing neighbor slides in Llama-31-8B (w/o Neighbor Slides) leads to a noticeable decrease in ROUGE-1 F1 (28.40 to 26.79) and ROUGE-2 F1 (11.68 to 10.72). Similar trends are observed in GPT-4o and Qwen2.5-7B, highlighting the importance of contextual information in maintaining logical coherence and reducing redundancy.

Balancing Full Context vs. Section Retrieval While using a section retriever helps reduce input length and improve efficiency, it can also cause minor variations in ROUGE scores. For example, Llama-31-8B with Section Retriever achieves slightly lower recall compared to its full-input counterpart. When provided with the full paper, they can better understand the broader context and underlying relationships, resulting in more accurate and coherent slide content. This suggests that LLMs have strong capabilities in processing long documents. Thus, in scenarios where the input length remains within the allowable range, feeding the full paper is often more advantageous for generating high-quality slides on a given topic.

However, in situations where the input length exceeds the model's context window or when the paper contains a significant amount of irrelevant information, **Section Retrieval** becomes essential. Selecting an optimal number of sections (e.g., 3 vs. 5) helps balance relevance and completeness. According to the results, **Proposed Method (5)** generally offers better recall and overall F1 compared to selecting fewer sections, as it provides more comprehensive contextual information without overwhelming the model with unnecessary details.

In summary, choosing between full-context input and section retrieval depends on the specific characteristics of the input paper. When the paper is relatively concise and highly relevant to the target topic, full-context input should be preferred. In contrast, for longer papers with diverse content,

Result Type	Element-Level	Slide-Level		Overall Impression	
	Align & Space	Logic	Coherence	Visual Appeal	Readability
Reference Slide	4.5	3.7	3.8	3.5	3.8
Baseline	2.0	3.0	3.3	2	2.5
JSON-Based Refinement	2.1	2.6	3.4	1.8	2.4
Our Method	3.0	3.8	3.4	2.8	3

Table 2: Evaluation results for layout generation

section retrieval is crucial for ensuring relevance while maintaining efficiency.

5.2 Factors Affecting Layout Quality

Alignment and Spacing metrics evaluate whether elements are properly positioned, evenly spaced, and free from overlap. As shown in Table 2, our method achieved a notable improvement in the Alignment and Spacing score (3.0) compared to the Baseline (2.0) and JSON-Based Refinement (2.1). Specifically, we observed that self-verification on JSON-based textual layout cannot improve the layout quality, even compromise the Logic, Visual Appeal, and Readability. Our method eliminates this problem and achieves consistent improvement by introducing the textual-to-visual modality transformation.

Taking a closer look at the wrong cases, the remaining problems fall into three types. (i) The quality of the initial layout plays a crucial role—severe errors, such as overlapping elements or inconsistent spacing, make it difficult for the Reviewer to provide accurate corrections. For instance, when multiple elements overlap, it becomes unclear which one should be adjusted. (ii) Additionally, the lack of diverse layout patterns in the training data, particularly for slides with images, limits the model's ability to position visual elements effectively. (iii) Finally, the complexity of multi-element layouts can cause small errors to propagate during refinement, leading to cascading issues that are challenging to resolve without advanced optimization strategies.

5.3 Complete Presentation Generation

While our current framework focuses on generating slides given a specific topic, the methodology can be naturally extended to automate the generation of a complete presentation composed of various slides.

Topic Generation and Slide Planning The first step in generating a full presentation is to extract key topics from the input paper. This can be achieved by analyzing the paper's structure (e.g., Abstract, Introduction, Method, Results). Additionally, keyword extraction and clustering techniques can help create a sequence of logically connected topics for the slides. Each generated topic corresponds to a unique slide.

Multi-Page Content Generation Once the topics are generated, the framework applies the content generation strategy iteratively for each slide. By incorporating context from the previously generated slides, the model maintains logical flow and coherence across the entire presentation. Special transition slides (e.g., Overview) can be inserted to improve the presentation's structure.

Consistent Layout and Visual Style The existing Reviewer + Refiner review process can be fully reused to ensure layout consistency across all slides.

This extension to full presentation generation holds significant practical value. It allows researchers to generate complete, high-quality presentations directly from academic papers, reducing the manual effort involved in slide creation.

6 Conclusion

In this paper, we propose a novel framework for generating academic presentation slides. By decomposing the task into content generation and layout generation, our method ensures adaptive layouts and visually consistent slides. We introduce a textual-to-visual iterative self-verification process using an LLM-based Reviewer + Refiner workflow, transforming complex textual layouts into visual representations for intuitive review and refinement. Experiments demonstrate that our approach significantly improves alignment, logical flow, visual appeal, and readability, offering a practical solution for automating high-quality slide generation.

Limitations

While our framework shows promising results in generating academic slides, it has two main limitations. First, the dataset is restricted to scientific papers and corresponding presentation slides from publicly available sources, which may limit its generalizability to other types of presentations. Second, the focus of our approach is primarily on generating accurate content and structured layouts, without considering advanced visual design aspects such as color schemes, animations, or aesthetic enhancements that contribute to overall slide polish and engagement.

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. In *Proceedings of the 17th International Natural Language Generation Conference*, pages 222–229, Tokyo, Japan. Association for Computational Linguistics.

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

Christopher Clark and Santosh Divvala. 2016. Pdffigures 2.0: Mining figures from research papers. In *Proceedings of the 16th ACM/IEEE-CS on Joint Conference on Digital Libraries*, JCDL '16, page 143–152, New York, NY, USA. Association for Computing Machinery.

Tsu-Jui Fu, William Yang Wang, Daniel J. McDuff, and Yale Song. 2021. Doc2ppt: Automatic presentation slides generation from scientific documents. In *AAAI Conference on Artificial Intelligence*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

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

Wenyi Hong, Weihan Wang, Qingsong Lv, Jiazheng Xu, Wenmeng Yu, Junhui Ji, Yan Wang, Zihan Wang, Yuxiao Dong, Ming Ding, et al. 2023. Cogagent: A visual language model for gui agents. *ArXiv preprint*, abs/2312.08914.

Yushi Hu, Weijia Shi, Xingyu Fu, Dan Roth, Mari Ostendorf, Luke Zettlemoyer, Noah A. Smith, and Ranjay Krishna. 2024. Visual sketchpad: Sketching as a visual chain of thought for multimodal language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Kermitt2. 2020. Grobid: Machine learning for extracting information from scholarly documents. `https://github.com/kermitt2/grobid`. Accessed: 2025-02-16.

Omar Khattab, Keshav Santhanam, Xiang Lisa Li, David Hall, Percy Liang, Christopher Potts, and Matei Zaharia. 2022. Demonstrate-search-predict: Composing retrieval and language models for knowledge-intensive nlp. *arXiv preprint arXiv:2212.14024*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020.

BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Chengzu Li, Wenshan Wu, Huanyu Zhang, Yan Xia, Shaoguang Mao, Li Dong, Ivan Vulić, and Furu Wei. 2025. Imagine while reasoning in space: Multimodal visualization-of-thought. *Preprint*, arXiv:2501.07542.

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.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual instruction tuning.

Ishani Mondal, Shwetha S, 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, St. Julian's, Malta. Association for Computational Linguistics.

OpenAI. 2023. Gpt-4 technical report. *ArXiv preprint*, abs/2303.08774.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. In *In the 36th Annual ACM Symposium on User Interface Software and Technol-*

ogy (UIST '23), UIST '23, New York, NY, USA. Association for Computing Machinery.

Simon Park, Abhishek Panigrahi, Yun Cheng, Dingli Yu, Anirudh Goyal, and Sanjeev Arora. 2025. Generalizing from simple to hard visual reasoning: Can we mitigate modality imbalance in vlms? *Preprint*, arXiv:2501.02669.

Zhenting Qi, Mingyuan Ma, Jiahang Xu, Li Lyna Zhang, Fan Yang, and Mao Yang. 2024. Mutual reasoning makes smaller llms stronger problem-solvers. *Preprint*, arXiv:2408.06195.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, dahai li, Zhiyuan Liu, and Maosong Sun. 2024. ToolLLM: Facilitating large language models to master 16000+ real-world APIs. In *The Twelfth International Conference on Learning Representations*.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Edward Sun, Yufang Hou, Dakuo Wang, Yunfeng Zhang, and Nancy X. R. Wang. 2021. D2S: Document-to-slide generation via query-based text summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1405–1418, Online. Association for Computational Linguistics.

Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen, Adam Pearce, Craig Citro, Emmanuel Ameisen, Andy Jones, Hoagy Cunningham, Nicholas L Turner, Callum McDougall, Monte MacDiarmid, C. Daniel Freeman, Theodore R. Sumers, Edward Rees, Joshua Batson, Adam Jermyn, Shan Carter, Chris Olah, and Tom Henighan. 2024. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. *Transformer Circuits Thread*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *ArXiv preprint*, abs/2307.09288.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024. Improving text embeddings with large language models. *Preprint*, arXiv:2401.00368.

Sida Wang, Xiaojun Wan, and Shikang Du. 2017. Phrase-based presentation slides generation for academic papers. In *AAAI Conference on Artificial Intelligence*.

Shengqiong Wu, Hao Fei, Leigang Qu, Wei Ji, and Tat-Seng Chua. 2023. Next-gpt: Any-to-any multimodal llm.

Wenshan Wu, Shaoguang Mao, Yadong Zhang, Yan Xia, Li Dong, Lei Cui, and Furu Wei. 2024. Mind's eye of LLMs: Visualization-of-thought elicits spatial reasoning in large language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Zhiheng Xi, Yiwen Ding, Wenxiang Chen, Boyang Hong, Honglin Guo, Junzhe Wang, Dingwen Yang, Chenyang Liao, Xin Guo, Wei He, Songyang Gao, Lu Chen, Rui Zheng, Yicheng Zou, Tao Gui, Qi Zhang, Xipeng Qiu, Xuanjing Huang, Zuxuan Wu, and Yu-Gang Jiang. 2024. Agentgym: Evolving large language model-based agents across diverse environments. *Preprint*, arXiv:2406.04151.

Guowei Xu, Peng Jin, Hao Li, Yibing Song, Lichao Sun, and Li Yuan. 2024. Llava-cot: Let vision language models reason step-by-step. *Preprint*, arXiv:2411.10440.

John Yang, Carlos E Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. 2024. Swe-agent: Agent-computer interfaces enable automated software engineering. *arXiv preprint arXiv:2405.15793*.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. 2022a. Webshop: Towards scalable real-world web interaction with grounded language agents. *Advances in Neural Information Processing Systems*, 35:20744–20757.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022b. ReAct: Synergizing reasoning and acting in language models. volume abs/2210.03629.

Zhuosheng Zhang, Aston Zhang, Mu Li, Hai Zhao, George Karypis, and Alex Smola. 2023. Multimodal chain-of-thought reasoning in language models. *arXiv preprint arXiv:2302.00923*.

Hao Zheng, Xinyan Guan, Hao Kong, Jia Zheng, Hongyu Lin, Yaojie Lu, Ben He, Xianpei Han, and Le Sun. 2025. Pptagent: Generating and evaluating presentations beyond text-to-slides. *arXiv preprint arXiv:2501.03936*.

A Detailed Descriptions of Reviewer and Refiner Modules

A.1 Reviewer Module

The Reviewer module simulates an expert evaluating the quality of a slide layout based on a set of predefined criteria. It analyzes the visual representation of the slide, identifies layout issues, and provides precise feedback for improvements. This feedback focuses on alignment, spacing, text overflow, and image distortion. The primary goal of the Reviewer is to detect errors and ensure that all elements are properly positioned and formatted for a visually coherent slide.

Evaluation Criteria and Feedback Rules:

Object Overlapping: Identifies overlapping elements and suggests repositioning or resizing to maintain separation between elements.

Image Quality and Distortion: Detects blurry or distorted images and recommends proportional scaling to enhance clarity.

Element Bounding and Text Overflow: Ensures text fits within its bounding box and suggests either expanding the box size or reducing font size.

Element Positioning and Alignment: Checks for consistent alignment and appropriate spacing between elements. Misaligned elements are adjusted to the nearest grid line.

Text Formatting Consistency: Verifies font family and text hierarchy, ensuring that title text is larger than body text.

Overall Composition and Visual Balance: Evaluates the slide's composition for symmetry and visual balance, recommending adjustments for better harmony.

Example Output:

```
[
  {"element": 302, "recommendation": "
      Increase text box height by 1.2x
      to fit overflowing text."},
  {"element": 303, "recommendation": "
      Move downward by 10% of its height
       to resolve overlap with ID
      302."},
  {"element": 304, "recommendation": "
      Reduce font size by 2pt to fit
      within the bounding box."}
]
```

A.2 Refiner Module

The Refiner module applies the Reviewer's feedback by modifying the structured layout described in JSON format. The task of the Refiner is to ensure that each adjustment improves the visual quality of the slide while maintaining the overall structure. This module focuses on correcting bounding box positions, resizing elements, and preventing overlaps.

The input to the Refiner consists of:

JSON File: Describes the position, size, font, and content of each element on the slide.

Reviewer's Feedback: Provides detailed recommendations for modifying elements (e.g., move, resize, align).

Slide Dimensions: Ensures all adjustments remain within the boundaries of the slide.

Modification Instructions:

Move an Element: Adjust the element's bounding box values to reposition it. Increase or decrease the top, bottom, left, and right values as required.

Resize or Scale an Element: Modify the width and height of an element proportionally while preserving its aspect ratio.

Avoid Overlap: Ensure no two elements overlap by repositioning or resizing conflicting elements.

Maintain Slide Boundaries: Prevent elements from exceeding the slide's width or height.

Example Input and Output:

Input JSON:

```
{
  "element": 302,
  "Bounds": [100, 200, 300, 400],
  "Font": {"size": 16},
  "Text": "Sample Text"
}
```

Refined Output:

```
{
  "element": 302,
  "Bounds": [100, 220, 300, 420],
  "Font": {"size": 14},
  "Text": "Sample Text"
}
```

By applying these refinements iteratively, the Refiner ensures that the final slide layout meets high visual and structural standards, resulting in an accurate and human-like output.

B Layout Evaluation Criteria and Scoring Standards

This section provides a detailed explanation of the evaluation criteria used to assess the quality of the generated slides. The evaluation process covers multiple aspects of slide design, including alignment, logical flow, text-visual consistency, visual appeal, and readability. Each criterion is scored on a five-point scale from 1 (Poor) to 5 (Excellent).

B.1 Alignment and Spacing

This criterion evaluates whether elements on the slide are properly positioned, evenly spaced, and free from overlap. It ensures that the layout maintains visual balance and clarity.

- **1 Point (Poor)**: Severe misalignment; text overlaps with visuals, creating a chaotic layout.
- **3 Points (Average)**: Most elements are aligned, but minor misplacements exist.
- **5 Points (Excellent)**: Perfect alignment and spacing with a professional layout.

Example Output:

```
{
  "reason": "Most elements are well-
      aligned, but the spacing between
      the title and body text is
      inconsistent.",
  "score": 4
}
```

B.2 Logical Flow

This criterion assesses the logical sequence of content, ensuring that the information presented in the slide is clear and structured for easy audience understanding.

- **1 Point (Poor)**: Disorganized content; key points do not follow a logical sequence.
- **3 Points (Average)**: Basic logical structure; minor reordering could improve the flow.
- **5 Points (Excellent)**: Seamless logical sequence with clear and structured information.

Example Output:

```
{
  "reason": "The information is
      structured logically, but the
      second point would be clearer if
      placed before the third.",
  "score": 4
}
```

B.3 Text-Visual Consistency

This criterion evaluates the consistency between text and visual elements such as images and charts. It ensures that visuals effectively support the textual information.

- **1 Point (Poor)**: Visuals are irrelevant or contradict the text.
- **3 Points (Average)**: Somewhat aligned, but better integration is needed.
- **5 Points (Excellent)**: Perfectly integrated visuals that reinforce the message.

Example Output:

```
{
  "reason": "The visuals effectively
      support the content, but the chart
       could be labeled more clearly.",
  "score": 4
}
```

B.4 Visual Appeal

This criterion assesses the overall aesthetic quality of the slide, focusing on color harmony, typography, and visual balance.

- **1 Point (Poor)**: Inconsistent styling; visually unappealing design.
- **3 Points (Average)**: Basic but functional color scheme; lacks enhancements.
- **5 Points (Excellent)**: Cohesive and visually appealing design with engaging elements.

Example Output:

```
{
  "reason": "The color scheme is
      visually appealing and harmonious,
       but the background contrasts too
      strongly with the text.",
  "score": 4
}
```

B.5 Readability

This criterion evaluates the readability and clarity of the text and graphical elements, ensuring that all content is easily understandable.

- **1 Point (Poor)**: Text is too small or has low contrast, making it unreadable.
- **3 Points (Average)**: Generally clear, but some areas need better contrast or spacing.
- **5 Points (Excellent)**: Highly readable with optimal font size, spacing, and contrast.

Example Output:

```
{
  "reason": "The text is clear, well-
      spaced, and maintains good
      contrast. The charts are easy to
      read and properly scaled.",
  "score": 5
}
```

These evaluation criteria ensure a comprehensive and structured assessment of the generated slides. By adhering to these standards, the evaluation process becomes interpretable, consistent, and reliable.