Slidester: Presentation Generation Using Generative AI

Hatim Mullajiwala
Artificial Intelligence & Machine Learning
Shri Dwarkadas J. Sanghvi College of Engineering, Mumbai, India

Haadi Rakhangi
Artificial Intelligence & Machine Learning
Shri Dwarkadas J. Sanghvi College of Engineering, Mumbai, India

Antara Kadam
Artificial Intelligence & Data Science
Shri Dwarkadas J. Sanghvi College of Engineering, Mumbai, India

Dr. Aruna Gawade
Artificial Intelligence & Machine Learning
Shri Dwarkadas J. Sanghvi College of Engineering, Mumbai, India

Dr. Nilesh Rathod
Artificial Intelligence & Machine Learning
Shri Dwarkadas J. Sanghvi College of Engineering, Mumbai, India

Laxmi Kurup
Artificial Intelligence & Data Science
Shri Dwarkadas J. Sanghvi College of Engineering, Mumbai, India

ABSTRACT

This study addresses the challenges of manual presentation creation in education, highlighting the limitations of traditional AI-driven approaches and introduces an innovative solution, "Slidester". Students across diverse academic domains, from science and technology to humanities and arts, often grapple with the time-intensive task of crafting multiple presentations, covering subjects such as scientific projects, literary analyses, and historical events. Slidester revolutionizes this process by leveraging two key methodologies: the integration of Large Language Models (LLMs), such as GPT, for tailored content creation, and the utilization of the web scraping techniques such as Tavily Web search API to access additional relevant content not covered by LLMs, meticulously refining it to align with the desired slide format. Recognizing the challenges in obtaining pertinent and high-fidelity images, crucial for educational presentations, Slidester initiates web searches for topic-related visuals as well as utilizes LCM-LoRA, a specialized Stable-Diffusion acceleration mechanism for image generation, ensuring a comprehensive representation of academic content. Furthermore, the system introduces a groundbreaking feature that addresses the time constraints faced by students—allowing the upload of documents to trigger Retrieval Augmented Generation (RAG). Beyond refining slide generation methodologies, Slidester introduces a user-centric feature: a personal assistant that streamlines interactive modifications to the presentation.

Keywords: Large Language Models, Automated Presentation, Retrieval Augmented Generation, Document-to-Presentation.

1. INTRODUCTION

In the realm of education, where the need for dynamic and up-to-date instructional materials is paramount, the traditional manual approach to crafting presentation slides proves to be both laborious and time-intensive. The painstaking process of manually selecting, arranging, and formatting educational content not only consumes valuable time but also introduces a higher likelihood of errors and inconsistencies. This manual method becomes particularly challenging when dealing with complex academic topics or frequent updates, as each modification requires meticulous attention to detail. The absence of automation in adjusting design elements not only hampers the efficiency of content creation but also diminishes the visual appeal of educational presentations, hindering the educator's ability to effectively convey information to students. As the demand for visually engaging and pedagogically effective presentations continues to grow within the educational landscape, the inefficiencies inherent in the manual slide creation process underscore the urgent need for advanced AI-driven solutions to revolutionize and streamline this crucial aspect of instructional communication.

In recent times, the advancement of Artificial Intelligence (AI) has notably accelerated, particularly within the realm of generative AI. This domain witnesses AI systems proficiently crafting content, ranging from emails, essays, and letters to intricate questions and beyond. Notably, an exemplary application of generative AI emerges in the realm of presentation slide creation. The prevailing market solutions and systems exhibit significant limitations, particularly in their ability to generate pertinent content for novel topics, especially when untrained by Large Language Models (LLMs). This constraint significantly hampers the possibilities for creating slides that align with cutting-edge information. Additionally, these systems grapple with the creation of contextually relevant images, often resulting in overly generalized visuals that lack diversity and fail to capture specific elements, such as architectural schematics or intricate flow charts.

Moreover, a notable deficiency in these systems lies in their incapacity to autonomously create presentations based on provided documents. For instance, the need to generate a presentation using a module outlined in a PDF file, adhering strictly to the content of the document, is not adequately addressed. Unfortunately, the current systems lack the functionality to upload such files and provide contextual information to guide the presentation creation process.

Furthermore, post-presentation creation, the existing systems fall short in allowing for autonomous adjustments to the slides through AI. This deficiency results in a time-consuming

manual process wherein adjustments, such as selecting text, adding or modifying content, adjusting font sizes on different slides, and managing color variations, must be carried out individually. This limitation not only hampers efficiency but also impedes the seamless integration of AI-driven adjustments, hindering the capacity to refine and customize presentations efficiently.

Introducing "Slidester", a pioneering system that adopts a tri-fold approach to transform AI-driven content generation for presentations.

1. The first method leverages renowned LLMs, such as GPT models, Llama2, and Wizard LLM. By benchmarking and selecting the most effective LLM based on outcomes, Slidester enhances content generation. Setting itself apart from existing systems, Slidester integrates Web search APIs like Google Serp API and Tavily, tapping into the latest online content to enhance the LLM's knowledge.

2. Next, Slidester introduces a fresh approach to image generation and retrieval. Many existing systems solely depend on image models to create images based on topics, often lacking the capability to capture specific image requirements or purposes. In contrast, Slidester's novel approach melds Web API integrations with specialized models such as LCM-LoRA. Utilizing this Latent Consistency Model (LCMs) LoRA diffusion module, Slidester not only streamlines image generation but also introduces a prompt interface. This feature allows users to specify image requirements or purposes, ensuring efficiency and precision based on provided textual cues.

3. Slidester's third approach tackles the task of creating slides tailored to specific documents. Designed for both educators and learners, this feature allows users to convert documents like project reports into engaging presentations. It achieves this using Retrieval Augmented Generation (RAG), a pre-trained seq2seq model. By harnessing RAG, Slidester amplifies the capabilities of LLMs, enhancing content generation by integrating external knowledge sources.

Beyond content creation prowess, Slidester encompasses an intuitive personal assistant, leveraging OpenAI's Assistant API. This feature facilitates dynamic slide editing, enabling users to seamlessly modify presentations by incorporating, deleting, or altering textual elements, images, fonts, colors, and a myriad of other presentation attributes.

2. RELATED WORK

In the research paper [1], Wizard LM presents Evol-Instruct, a novel methodology utilizing an evolutionary algorithm to generate detailed instructions for Large Language Models (LLMs). The study demonstrates the efficacy of this approach, showcasing improved LLM performance in complex tasks and competitive scores across different domains. However, the authors acknowledge limitations, particularly the context-specific nature of their evaluations using automatic GPT-4 and human assessments, as well as potential gaps in test scenario coverage. Notably, the paper underscores the ethical and societal implications of AI-generated instructions, emphasizing the need for further research in this area. In essence, Wizard LM's work contributes to advancing LLM performance by introducing innovative instruction data creation methods [1].

The research paper [2] systematically investigates the natural language processing (NLP) capabilities of the GPT-3 and GPT-3.5 series models, assessing their performance across nine natural language understanding (NLU) tasks using 21 datasets. Despite the introduction of the RLHF training strategy, the study reveals that the overall proficiency of GPT series models in NLU tasks does not show a consistent improvement with model evolution. While enhancing the ability to generate human-like responses, the RLHF strategy compromises performance on specific tasks, underscoring the need for further refinement, especially in terms of model robustness. The paper acknowledges limitations, including dataset constraints due to OpenAI API restrictions and the unavailability of the GPT-4 API for comparison. The authors highlight the crucial importance of investigating GPT-4 capabilities for future research, providing valuable insights into the strengths, limitations, and potential improvements in the natural language processing domain [2].

In their paper titled "Principled Instructions Are All You Need for Questioning LLaMA-1/2, GPT-3.5/4" [3], Bsharat et al. present a set of 26 guiding principles designed to optimize the interrogation process of large language models, specifically LLaMA-1/2 and GPT-3.5/4. These principles cover aspects ranging from conciseness and clarity to user interaction and engagement, providing a comprehensive framework for refining user prompts. Through rigorous experimentation, the authors validate the efficacy of these principles, demonstrating their capacity to elicit high-quality responses from pre-trained large language models. This work contributes valuable insights into improving interactions with such models, offering a pertinent reference for enhancing the effectiveness of large language model applications [3].

The paper "LCM-LoRA: Universal Stable-Diffusion Acceleration Module for Text-to-Image Generation" delves into the utilization of Latent Consistency Models (LCMs) and LoRA

distillation to expedite text-to-image generative tasks. Demonstrating that LCM-LoRA significantly enhances image generation quality while economizing on memory consumption, the authors position it as a versatile accelerator applicable across diverse image generation scenarios. Comparative analysis with previous numerical PF-ODE solvers underscores LCM-LoRA's superiority in terms of both speed and quality. In essence, this paper introduces a promising strategy for boosting the efficiency and efficacy of text-to-image generation models [4].

The paper "High-Resolution Image Synthesis with Latent Diffusion Models" introduces a groundbreaking methodology for image synthesis employing Latent Diffusion Models (LDMs) [5]. Leveraging a diffusion process, these generative models prove capable of producing high-quality images from noise vectors, exhibiting state-of-the-art quality and diversity when trained on large-scale datasets. The authors further enhance the LDM framework by incorporating novel strategies such as training in the latent space of pretrained autoencoders, rescaling the latent space for improved signal-to-noise ratio, and employing classifier-free guidance to enhance visual fidelity. Demonstrating the versatility of LDMs, the paper showcases their ability to generate images based on various conditioning inputs like text or bounding boxes, thereby facilitating tasks such as layout-to-image synthesis, class-conditional image synthesis, object removal, and superresolution. In essence, this paper offers a comprehensive and innovative approach to high-resolution image synthesis, achieving state-of-the-art results across a diverse range of tasks [5].

The paper "Adversarial Diffusion Distillation (ADD)" introduces an innovative training approach for generative models, facilitating high-quality image synthesis in a few steps [6]. The ADD framework comprises a generator and a discriminator, both trained using a combination of adversarial loss and a novel technique called score distillation. This involves aligning the generator's scores with those of a pre-trained reference model, enhancing the quality and diversity of the generated images. The authors demonstrate the efficacy of ADD across benchmark datasets, achieving state-of-the-art results measured by quantitative metrics (FID and CLIP score) and human preference evaluations. An ablation study further investigates the impact of different design choices on ADD's performance, affirming its promising potential for fast and efficient image synthesis. The implications extend to various domains, including computer vision, graphics, and creative AI [6].

The paper "Automatic Slide Generation for Scientific Papers" [7] introduces a pioneering hierarchical sequence-to-sequence approach for the automatic generation of presentation slides from scientific documents. Incorporating document summarization, image and text retrieval, and slide structure, the proposed method surpasses strong baselines, yielding slides with enriched content and aligned imagery. The release of a dataset containing approximately 6,000 paired

documents and slide decks facilitates accelerated research in this domain. Noteworthy improvements include the integration of a paraphrasing module to enhance the quality of textual content and a user study validating the perceived quality of the generated slides. This work significantly contributes to vision-and-language understanding, addressing the unique challenges of document-to-slide generation, and introduces a valuable dataset and metrics to advance the state-of-the-art in this field. The discussion on human-AI collaboration highlights the practical relevance of the proposed approach, offering opportunities for efficient content creation and review [7].

The paper introduces Retrieval-Augmented Generation (RAG) as a fine-tuning recipe for language models, combining pre-trained parametric and non-parametric memory to enhance language generation [8]. Focusing on knowledge-intensive Natural Language Processing (NLP) tasks, the authors showcase the effectiveness of RAG models in fact verification, question answering, and question generation tasks. Comparative analyses against other state-of-the-art models reveal the competitive performance of RAG models without the need for complex pipeline systems or domain-specific architectures. Addressing the limitations of large pre-trained language models in knowledge manipulation, the paper positions RAG models as a viable solution. This work represents a significant stride in advancing language models' capacity to access and manipulate knowledge for NLP tasks, offering a promising avenue for further exploration in the field [8].

III. METHODOLOGY

3.1. Model Selection:

Employing a large language model (LLM), the proposed method generates information tailored to individual slides within a presentation. This information is integrated with images acquired through web scraping. Additionally, the methodology integrates an image generation model and a text embedding model. The former creates images not attainable through web scraping, while the latter establishes a vector database from a document. The image generation model is accessible through a virtual assistant, capable of processing textual or voice prompts, as elaborated in subsequent sections.

3.1.1. Large Language Model:

The crux of this approach lies in the meticulous selection of a suitable Large Language Model (LLM), given its direct impact on the accuracy, speed, reliability, and utility of the generated information. This selection process adheres to key criteria crucial for optimal performance. Factors considered include the factual correctness of generated text, the model's propensity for

delivering harmless, honest, and helpful responses, its natural language understanding capabilities, knowledge across diverse domains, LLM size in terms of parameters, supported context length, hardware requirements, and efficient inference time. The evaluation of various LLMs based on these criteria is detailed in the table below.

MODELS	HellaSwag	MMLU	GSM8K	#Params(B)
Mistral-7B	83.31	64.16	37.83	7.24
WizardLM-13B	82.21	54.64	13.5	13
GPT 4 Turbo (1106 preview)	95.3	69.9	66.8	1760
GPT 3.5 Turbo(0613)	85.5	61.4	50.1	154
Llama 2(70B)	85.3	68.9	56.7	70

Fig 3.1.1: LLM Comparison on various Benchmarks

3.1.2. Image Generation Model:

Conditional text-to-image generation and Latent Diffusion models prove practical in this method. When selecting an image generation model, factors such as the generated image's relevance to the prompt, quality, supported domains, inference time, and required hardware should be considered. Various stable diffusion models from HuggingFace have been rigorously tested based on these criteria. The evaluation indicated that Stable Diffusion XL Base 1.0 by StabilityAI performed effectively in this approach. Comprising a base component generating noisy latents and a refinement model denoising them, this model includes two pretrained text encoders, OpenCLIP-ViT/G and CLIP-ViT/L, which align text and image domains. Despite requiring 40-50 inference steps, this state-of-the-art model surpassed prior diffusion variants in user preference.

To enhance the efficiency of the image generation, a focus of paramount importance within the overarching presentation generation process, integration of Latent Consistency Model-Latent Residual Adapters (LCM-LoRA) or Adversarial Diffusion Distillation (ADD) can be considered. LCM-LoRA, when combined with SDXL Base 1.0 or SDXL-Turbo utilizing Adversarial Diffusion Distillation, is identified as a fitting model tandem for this methodology. SDXL Latent Consistency Model (LCM), as proposed in “Latent Consistency Models: Synthesizing High-Resolution Images with Few-Step Inference,” revolutionizes the image generation process by reducing the number of steps required. It distills the original SDXL model into a version that needs fewer steps (4 to 8 instead of 25 to 50) for image generation. This model is particularly advantageous for applications requiring faster image generation without compromising quality. It is noteworthy for being 50% smaller and 60% faster than the original SDXL.Additionally, SDXL Turbo is a newly released variant of SDXL 1.0, developed for “real-time synthesis.” This means it can generate images extremely quickly, a feature powered by a new training method called

Adversarial Diffusion Distillation (ADD). This variant is unique due to its lossy autoencoding component, which, although results in some loss of information during the encoding and decoding of images, enables faster image generation. The graph below illustrates a comparison between the SDXL-Turbo model and other image generation models across varying numbers of inference steps, sourced from reference [31].

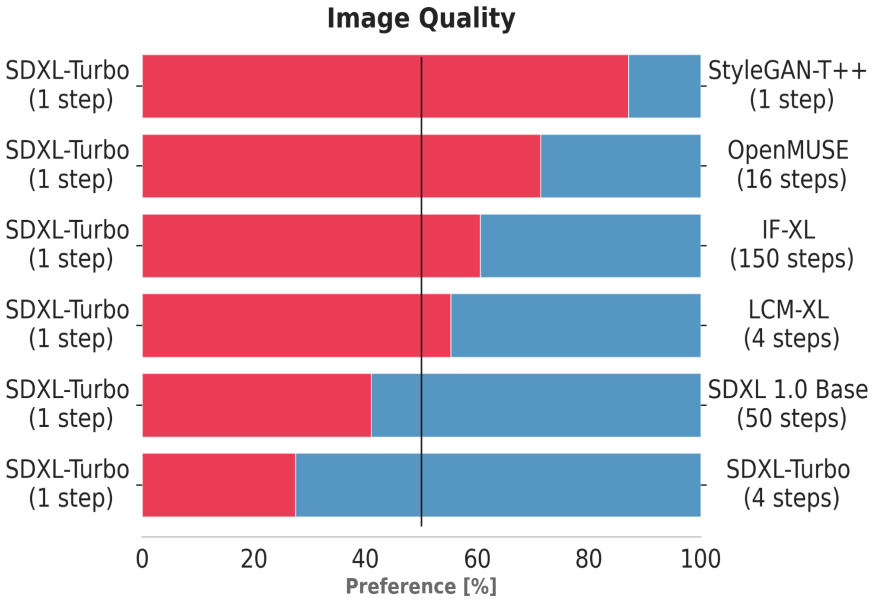

Fig 3.1.2.1: SDXL Turbo Image Quality Comparison

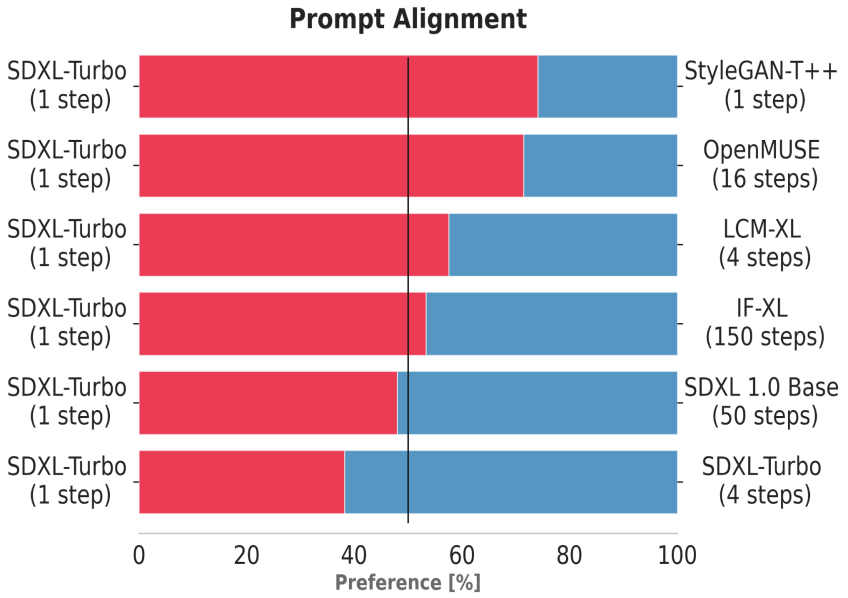

Fig 3.1.2.2: SDXL Turbo Image Prompt Alignment

3.1.3 Text Embedding Model:

The text embedding model plays a crucial role in the outlined methodology as it directly influences the quality of the embeddings generated from the user document. This, in turn, has an impact on retrieval through similarity search methods like cosine similarity. Key criteria to consider when selecting an appropriate text embedding model include the model's size, the size of the embeddings it generates, and the corresponding time required to create embeddings from a document. The MTEB leaderboard on HuggingFace provides a valuable resource for comparing and selecting text embedding models that align with the aforementioned criteria.

3.2. Architecture:

3.2.1. Basic Presentation Generation Pipeline:

The process begins by selecting a specific domain and a topic for the presentation. Following this, a designated Large Language Model (LLM) is utilized to generate a set of relevant slide titles, providing recommendations for potential inclusions aligned with the specified presentation title. Flexibility is maintained in customizing these suggestions by removing, adding, or rearranging titles, and by specifying the desired quantity of information points for each slide. The confirmed titles are then input into the LLM, which generates comprehensive point information for each slide based on the specified number of points.

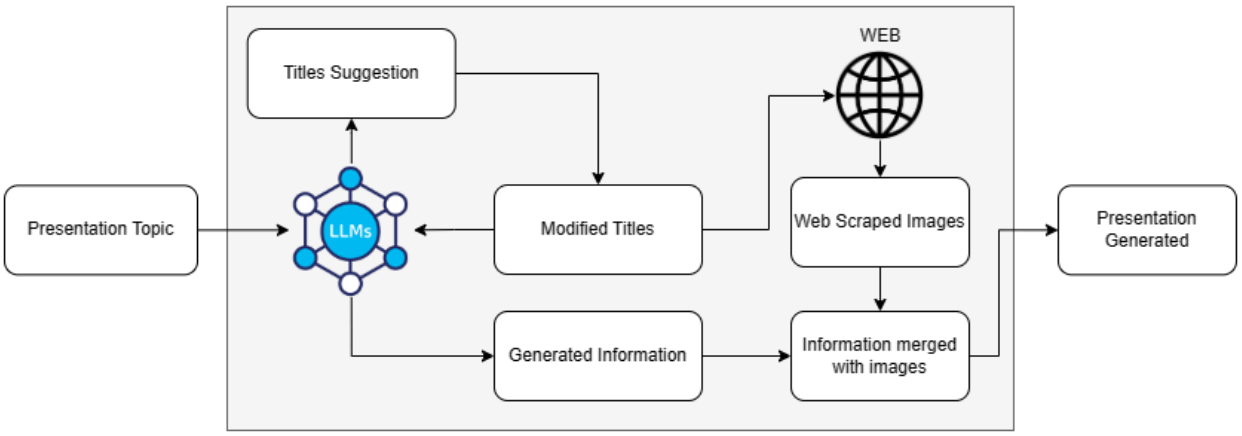

Fig 3.2.1. Generation Pipeline

The procedure incorporates web scraping to identify pertinent images for each slide, with options including the utilization of Google SerpAPI, Tavily, or manual methods. Furthermore, an option to select an image from the pool of web-scraped images is integrated. The textual content and chosen images are seamlessly merged, resulting in a comprehensive presentation that harmoniously combines thoughtfully crafted content with visually engaging elements.

Algorithm 1: Basic presentation generation pipeline

Input:

1. D: Domain selected by User.
2. T: Topic specified by User.
3. M: Selected Large Language Model

Output: Presentation Slides with text and images

1. **Domain & Topic Selection:**
   read(D)
   read(T)

2. **Slide Title Generation**:
   LLM = initialize_llm(M)
   slide_titles = LLM.generate.slide_titles(D,T)

3. **Slide Title Customization:**
   num_points = get_num_points_from_user(title)
   titles_order = get_titles_order_from_user()
   confirmed_titles = customize_titles(slide_titles, num_points, titles_order)

4. **Slide point Information Generation**
   Initialize empty dictionary slide_points
   For each title in confirmed_titles **Do:**
   num_points = get_points(title)
   slide_points[title] = LLM.generate_point_information(title, num_points)
   End For

5. **Web Scraping for Images:**
   Initialize empty dictionary selected_images
   For each title in confirmed_titles **Do:**
   search_query_for_image = generate_search_query()
   images = web_scraped_images(search_query)
   selected_images[title] = select_image(images)
   End For

6. **Merge generated text and images:**
   Initialize empty dictionary presentation_slides
   For each title in confirmed_titles **Do:**
   text_content = slide_points[title]
   image = selected_images [title]
   presentation_slides[title] = merge_text_and_images(title, text_content, image)

   End For

7. **Display generated Presentation:**
   display_presentation(presenation_slides)

8. **End**

3.2.2. Web-enabled Presentation Generation Pipeline:

Large Language Models (LLMs) are trained on limited data, resulting in their knowledge being constrained to a specific point in time. This limitation is often referred to as LLMs being "stuck in time," leading to potential issues such as the generation of factually incorrect information or hallucinations, particularly concerning new or the topics the language model has not been trained on. While one option to address this is fine-tuning the LLM on new data, it is computationally inefficient and unoptimized, especially considering the diverse range of domains the model may be unaware of.

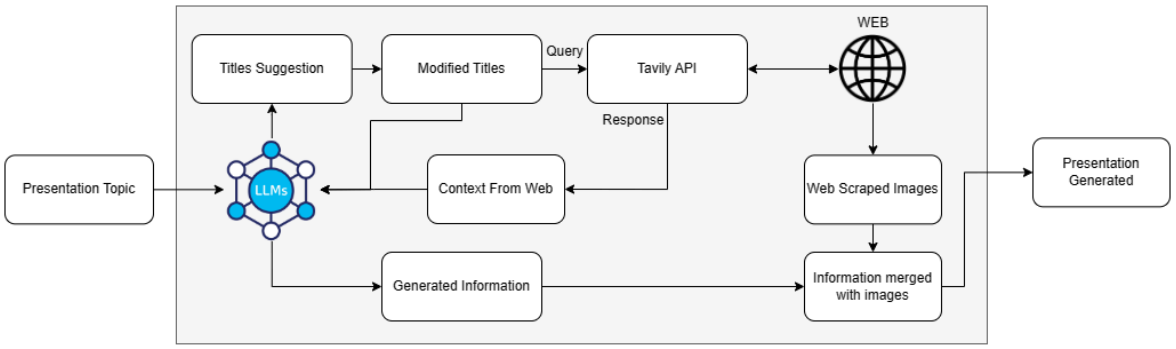

Fig 3.2.2: Web Generation Pipeline

Algorithm 2: Web-Enabled Presentation Generation Pipeline

Input:

1. D: Domain selected by User.
2. T: Topic specified by User.
3. M: Selected Large Language Model

Output: Presentation slides with enriched information from the web

1. **Domain & Topic Selection:**
   read(D)
   read(T)

2. **Slide Title Generation**:
   LLM = initialize_llm(M)
   web_context = retrieve_web_context(T)
   slide_titles = LLM.generate.slide_titles_from_context(D, T,web_context)

3. **Slide Title Customization:**
   num_points = get_num_points_from_user(title)
   titles_order = get_titles_order_from_user()
   confirmed_titles = customize_titles(slide_titles, num_points, titles_order)

4. **Slide Point Information Generation with Web Context:**
   Initialize empty dictionary slide_points
   For each title in confirmed_titles **Do:**
   web_context = retrieve_web_context(title)
   num_points = specify_points(title)
   slide_points[title] = LLM.generate_slide_points_with_context(title, num_points, web_context)
   End For

5. **Web Scraping for Images:**
   Initialize empty dictionary selected_images
   For each title in confirmed_titles **Do:**
   search_query_for_image = generate_search_query()
   images = web_scraped_images(search_query)
   selected_images[title] = select_image(images)
   End For

6. **Merge generated text and images:**
   Initialize empty dictionary presentation_slides
   For each title in confirmed_titles **Do:**
   text_content = slide_points[title]
   image = selected_images [title]
   presentation_slides[title] = merge_text_and_images(title, text_content, image)

   End For

7. **Display generated Presentation:**
   display_presentation(presenation_slides)

8. **End**

An optimized approach would be enriching the knowledge of the LLM by adding additional information about the topic in the context of the LLM from web-sources. This knowledge enrichment, when combined with the natural language understanding capabilities of the LLM, can facilitate the LLM to generate information on nearly any topic even if the model has limited or no knowledge on the given topic. This methodology leverages Tavily's search API, a specialized engine designed specifically for LLMs. This API provides real-time and factual information, augmenting the presentation content with updated and relevant details, even in cases where the model has limited or no prior knowledge of the given topic.

Within the web-enabled presentation generation pipeline, the procedure commences similarly to the basic pipeline by selecting the domain and topic for the presentation. Subsequently, this

information is conveyed to the Tavily Search API, which enhances the LLM's knowledge, suggesting titles in alignment with the presentation topic. The information for each slide is then generated by retrieving and incorporating relevant data from the web into the LLM's context. Subsequently, the LLM utilizes this enriched context to generate information corresponding to the specified number of points. Once again, this information is amalgamated with images to produce the final presentation.

3.2.3. Document-Specific Presentation Generation Pipeline:

The document-specific presentation generation pipeline employs a Retrieval Augmented Generation (RAG) based approach. This entails constructing a vector database from the provided document, which may be in docx, ppt, txt, or csv format, utilizing the designated text embedding model. The resulting vector database becomes pivotal for enriching the presentation with the information encapsulated in the document. To facilitate this enrichment, a similarity search is conducted using methods such as cosine similarity, enabling the retrieval of information pertinent to a specific slide.

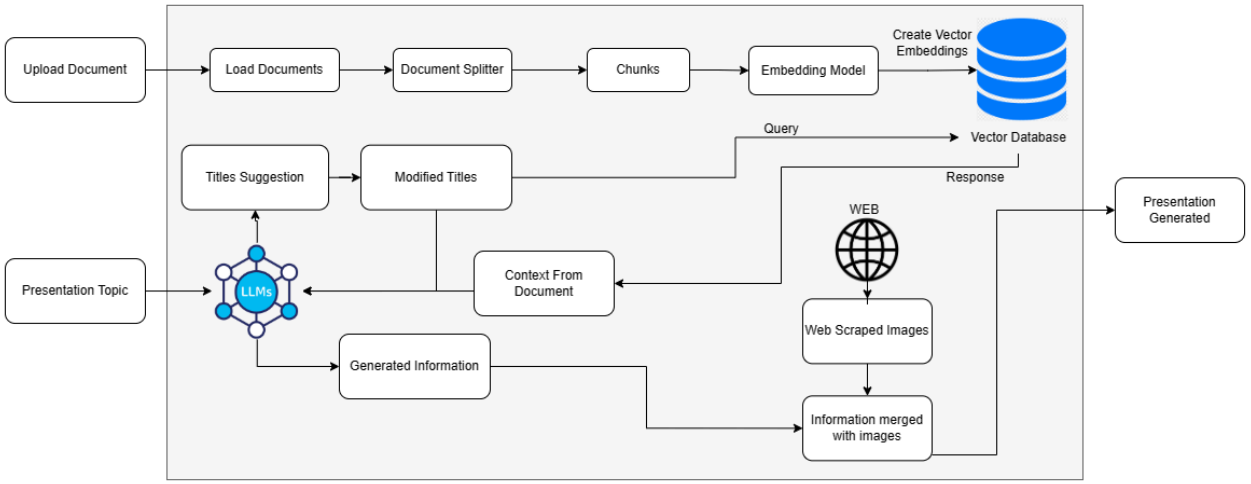

Fig 3.2.3: Document-Specific Generation Pipeline

Algorithm 3: Document-Specific Presentation Generation Pipeline

Input:

1. P: Path to the document file.
2. E: Text embedding model.
3. D: Domain selected by User.
4. T: Topic specified by User.
5. M: Selected Large Language Model

Output: Presentation slides enriched with information from the document

1. **Domain, Topic & User Document Input:**
   read(D)
   read(T)
   read(P)

2. **Document Vectorization:**
   text = read_document(P)
   doc_embeddings = create_embeddings(text, E)
   vectorstore = create_vectorstore(doc_embeddings)

3. **Slide Title Generation**:
   k →number of relevant document to return
   LLM = initialize_llm(M)
   relevant_context = vectorstore.similarity_search(T, k)
   slide_titles = LLM.generate.slide_titles_from_context(D, T, relevant_context)

4. **Slide Title Customization:**
   num_points = get_num_points_from_user(title)
   titles_order = get_titles_order_from_user()
   confirmed_titles = customize_titles(slide_titles, num_points, titles_order)

5. **Generate Document-Specific Presentations:**
   k →number of relevant document to return
   Initialize empty dictionary slide_points
   For each title in confirmed_titles **Do:**
   relevant_context = vectorstore.similarity_search(title, k)
   num_points = specify_points(title)
   slide_points[title] = LLM.generate_slide_points_with_context(title, num_points, relevant_context)
   End For

6. **Web Scraping for Images:**
   Initialize empty dictionary selected_images
   For each title in confirmed_titles **Do:**
   search_query_for_image = generate_search_query()
   images = web_scraped_images(search_query)
   selected_images[title] = select_image(images)
   End For

7. **Merge generated text and images:**
   Initialize empty dictionary presentation_slides
   For each title in confirmed_titles **Do:**
   text_content = slide_points[title]
   image = selected_images[title]
   presentation_slides[title] = merge_text_and_images(title, text_content, image)

   End For

8. **Display generated Presentation:**
   display_presentation(presenation_slides)

9. **End**

In this workflow, the slide title functions as a query for the similarity search. The retrieved document then serves as context for the Large Language Model (LLM). Subsequently, the LLM harnesses this contextual information to generate document-specific presentations. The size of the embeddings created plays a pivotal role in this process. A larger embedding size contributes to a more robust vector representation of the document, enhancing the retrieval of the most relevant information. In instances where the input document is relatively small or excessively large, there is a possibility of the model generating information irrelevant to a slide. This challenge can be

addressed by iteratively refining prompts to utilize only the most pertinent context returned by the similarity search.

3.2.4. Virtual Assistant Powered Presentation Modification:

The proposed system advocates for the utilization of a virtual assistant to integrate functionalities such as adjusting slide information, generating new slides, or removing slides as needed, along with creating visually appealing images not sourced from the internet using the selected image generation model. Furthermore, users have the capability to customize presentation elements like font, style, and color. These functionalities are accessible through voice or text prompts to the assistant, facilitating natural conversation-based modifications. The OpenAI Assistant API, offering a range of tools for tasks like code interpretation, knowledge retrieval, and function calling, has proven to be suitable for this purpose. Alternatively, open-source alternatives like Langchain's Opengpts, providing similar and additional functionalities, can also be employed for the virtual assistant.

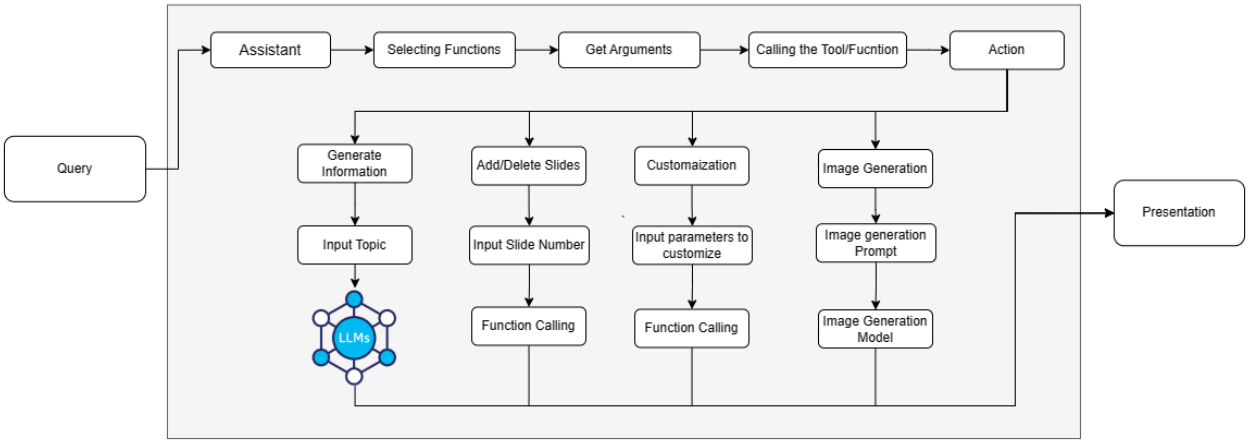

Fig 3.2.3: Assistant Pipeline

Function calling, where the model generates JSON objects with arguments to invoke a developer-defined function, is pivotal in this process, facilitating direct interaction with the image generation and the LLM. The model is trained to discern the relevant function based on user prompts, execute the identified function, and return an output JSON containing necessary task arguments.

In tasks such as generating or adding information on a specific topic, the model necessitates details like the topic, slide number, and the number of information points to be added. Deleting a slide is accomplished by providing the slide number. In the context of image generation, the model comprehends the user's intent through conversation, generates a query, and transmits it to the image generation model. The model identifies the given parameters, determines which functions to call in all of these scenarios, and returns a JSON object with the function name and

arguments, facilitating the execution of specific tasks. Furthermore, the virtual assistant can also generate charts if a CSV file is provided as input in the document-specific pipeline, enhancing the platform's versatility and utility.

Algorithm 4: Virtual Assistant Powered Presentation Modification

Input:

1. P: Presentation
2. I: Image generation Model
3. U: User Query
4. A: Open AI Assistant
5. M: Selected Large Language Model

Output: Modified presentation with adjusted information and visually appealing images

1. **Get User query:**
   read(U)

2. **Assistant Interactions:**
   Initialize a list of pre-defined functions assistant_functions.
   A = initialize_assistant()
   LLM = initialize_llm(M)
   function_to_call = LLM.get_function_from_query (U, assistant_functions)

3. **Function Calling for Presentation Modification:**
   If function_to_call == add_slide **Then**
   params = function_to_call.get_params()
   new_presentation = add_slide(params, P)
   End
   If function_to_call == remove_slide **Then**
   params = function_to_call.get_params()
   new_presentation = remove_slide(params, P)
   End
   If function_to_call == adjust_text **Then**
   params = function_to_call.get_params()
   new_presentation = adjust_text(params, P)
   End
   If function_to_call == customize_font **Then**
   params = function_to_call.get_params()
   new_presentation = customize_font (params,P)
   End
   If function_to_call == generate_image **Then**
   params = function_to_call.get_params()
   image = generate_image(I, params)
   new_presentation = add_generated_image(image,P)
   End
   If function_to_call == generate_chart **Then**
   csv = get_csv(U)
   params = function_to_call.get_params()
   chart = make_chart(csv, params)
   new_presentation = add_generated_chart(chart, P)
   End

4. **Display Presentation given by Assistant:**
   display_presentation_by_assistant(new_presentation)

5. **End**

IV. RESULTS

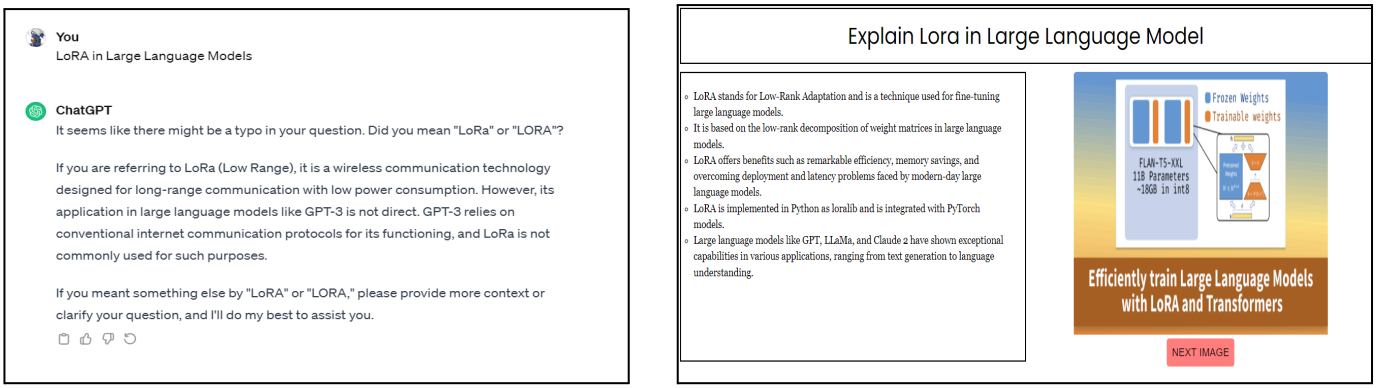

Fig 4.1: Web Context Comparison

The findings of the web-enabled presentation generation pipeline are delineated herein. A comparative analysis is conducted between Slidester's methodology and the GPT-3.5-turbo model developed by OpenAI, trained on data up to September 2021. The outcomes reveal that the GPT-3.5-turbo model, when utilized in isolation, exhibits a lack of awareness regarding posed queries. However, when integrated into Slidester's Web-enabled presentation generation pipeline, the model demonstrates the ability to produce engaging slides incorporating pertinent information gleaned from the internet.

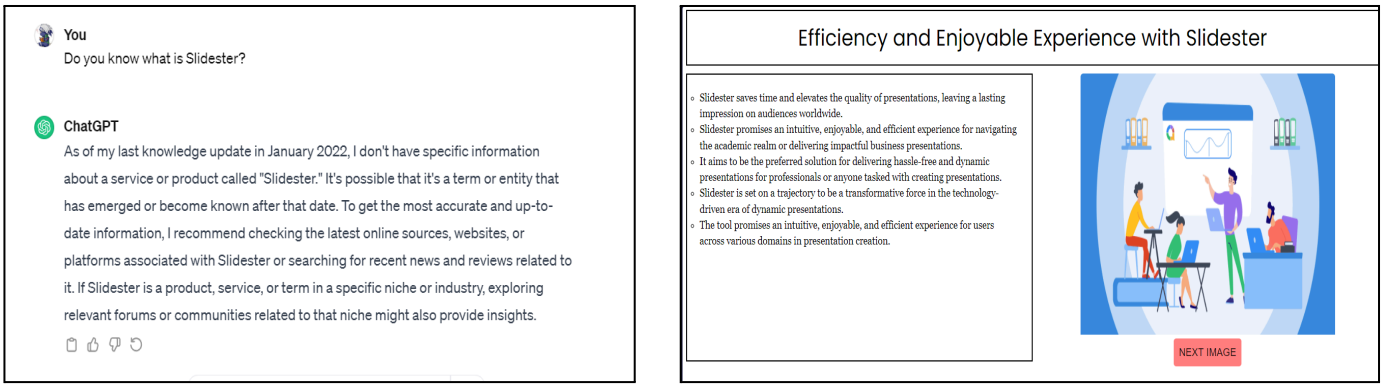

Fig 4.2: User Document Context Comparison

Despite being endowed with information from the web, any Generative AI model inherently lacks access to the user's private documents. However, leveraging the RAG (Retrieval-Augmented Generation) based document-specific presentation generation pipeline developed by Slidester facilitates the creation of compelling presentations rooted in the user's personal data. This innovative approach holds potential for diverse applications, offering a valuable tool for the generation of presentations across different business sectors and organizations. The aforementioned instance showcases the creation of a presentation focused on the research paper

titled 'Slidester: Presentation Generation Using Generative AI,' achieved by supplying the document for processing.

System	Document Relevancy %	Web Relevancy %	Factual Correctness %	Image diversity %
Manually	90	95	99	90
Existing AI System	Not supported	80	50	70
Slidester	95	95	90	95

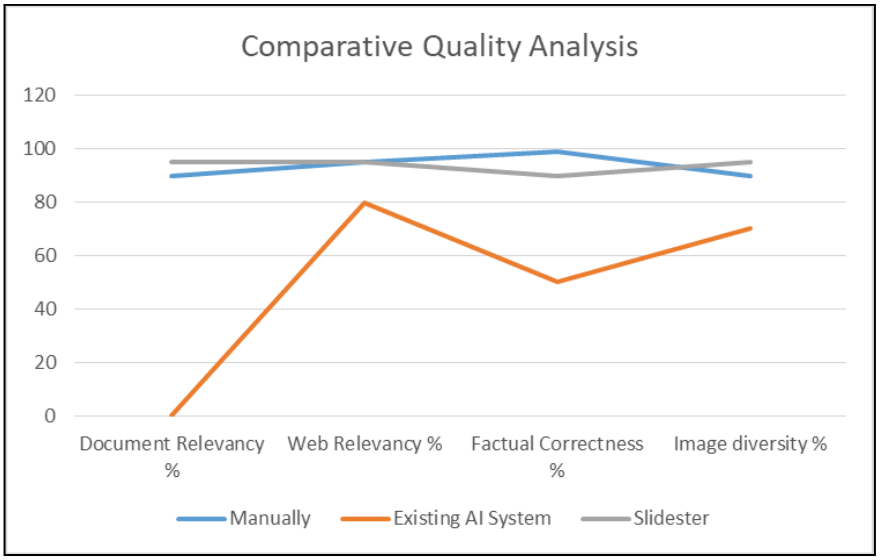

Fig 4.3: Comparative Analysis of Quality: Existing System VS Slidester

The chart above depicts a comparative analysis encompassing various factors, including relevancy to the provided document, relevance to the web, factual accuracy of the presentation, and diversity in image selection. While manual methods exhibit superior performance across these factors, it is noteworthy that they demand a substantial investment of time and resources, as evidenced by subsequent charts. In contrast, existing AI systems face limitations in accommodating user documents as input and accessing pertinent information from the web, particularly for contemporary subjects. Slidester, benefiting from its diverse presentation generation approaches, excels in this context. Notably, it achieves heightened image diversity by enabling the inclusion of images through web scraping and leveraging a text-to-image generation model.

System	Research(in mins)	Add information(in mins)	Add Images(in mins)	Do Modifications (in mins)	Overall Time(5 slides)
Manually	60	20	20	40	140
Existing AI System	5	5	10	20	40
Slidester	3	10	15	5	33

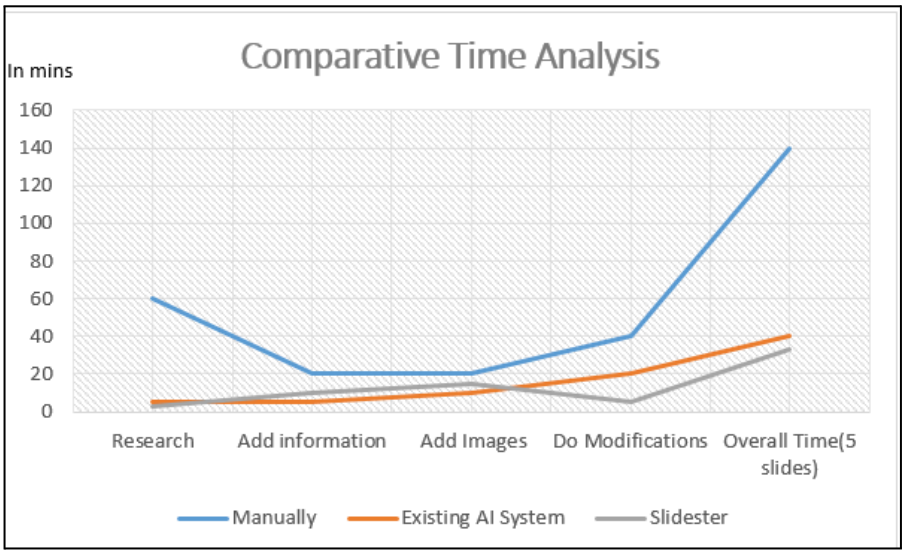

Fig 4.4: Comparative Analysis of Time: Existing System VS Slidester

The chart presented above provides a comparative evaluation of time efficiency in creating presentations (consisting of 5 slides) across manual methodologies, existing AI systems, and Slidester's innovative approach. The overall time allocation for presentation creation is delineated into three categories: time spent on researching the topic, time dedicated to sourcing and incorporating pertinent information, and time invested in adding images and effecting modifications to the presentation. Notably, Slidester demonstrates substantial time savings in each of these aspects compared to the manual creation of presentations, with manual methodologies being referenced from a website whose details can be found at [32].

In comparison to existing AI systems, Slidester exhibits superior performance, particularly excelling in research and presentation modification. This efficacy is attributed to Slidester's streamlined presentation pipeline, requiring only a user-provided input topic, and the inclusion of a virtual assistant that facilitates effortless editing and modification through straightforward textual or voice prompts.

V. CONCLUSION

In conclusion, the proposed presentation generation system, driven by a versatile virtual assistant, presents a comprehensive and innovative approach, particularly relevant for educational purposes. By seamlessly integrating functionalities for adjusting slide information, generating new slides, customizing presentation elements, and creating visually appealing images, the system caters to diverse user needs within the educational landscape. The utilization of OpenAI's Assistant API, along with potential alternatives like Langchain's opengpts, underscores the flexibility of the system, making it a valuable asset for educational content creation. Through natural

conversation-based prompts, users, including educators and students, can dynamically modify content, benefiting from the Assistant API's streamlined conversation management and function calling capabilities.

The document-specific presentation generation pipeline, employing a Retrieval Augmented Generation (RAG) approach, enhances adaptability by constructing a vector database from user-provided documents. This vector database enables the system to retrieve and incorporate relevant information into presentations, showcasing a nuanced understanding of document context, a feature particularly beneficial in educational settings.

Furthermore, the incorporation of web-enabled features, powered by Tavily's search API, enriches the system's knowledge base in real-time, addressing the limitation of large language models being "stuck in time." By seamlessly blending information retrieval from the web with natural language understanding capabilities, the system ensures accurate and up-to-date content generation across a myriad of educational topics.

The virtual assistant's ability to interact through voice or text prompts, coupled with the integration of image generation models and customizable elements, enhances user engagement and facilitates dynamic content creation in educational contexts. Leveraging Threads and Messages, the system maintains a coherent conversation history, enhancing the model's contextual understanding, which is crucial for educational interactions.

In essence, this presentation generation system represents a synthesis of cutting-edge technologies, providing educators and learners with a powerful tool for creating dynamic, customized, and visually appealing presentations. The convergence of retrieval augmented generation, web-enabled features, and a versatile virtual assistant marks a significant stride towards efficient, adaptable, and user-friendly content creation in the realm of educational presentations.

VI. REFERENCES

[1] Xu, Can, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin Jiang. “WizardLM: Empowering Large Language Models to Follow Complex Instructions.” arXiv.org, June 10, 2023. https://arxiv.org/abs/2304.12244.

[2] Ye, Junjie, Xuanting Chen, Nuo Xu, Can Zu, Zekai Shao, Shichun Liu, Yuhan Cui, et al. “A Comprehensive Capability Analysis of GPT-3 and GPT-3.5 Series Models.” arXiv.org, December 23, 2023. https://arxiv.org/abs/2303.10420.

[3] Bsharat, Sondos Mahmoud, Aidar Myrzakhan, and Zhiqiang Shen. “Principled Instructions Are All You Need for Questioning Llama-1/2, GPT-3.5/4.” arXiv.org, December 26, 2023. https://arxiv.org/abs/2312.16171.

[4] Luo, Simian, Yiqin Tan, Suraj Patil, Daniel Gu, Patrick von Platen, Apolinário Passos, Longbo Huang, Jian Li, and Hang Zhao. “LCM-Lora: A Universal Stable-Diffusion Acceleration Module.” arXiv.org, November 9, 2023. https://arxiv.org/abs/2311.05556.

[5] Rombach, Robin, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Bjorn Ommer. “High-Resolution Image Synthesis with Latent Diffusion Models.” 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022. https://doi.org/10.1109/cvpr52688.2022.01042.

[6] Sauer, Axel, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. “Adversarial Diffusion Distillation.” arXiv.org, November 28, 2023. https://arxiv.org/abs/2311.17042.

[7] Tsu-Jui, William Yang Wang, Fu, Daniel McDuff, and Yale Song. “DOC2PPT: Automatic Presentation Slides Generation from Scientific Documents.” Proceedings of the AAAI Conference on Artificial Intelligence 36, no. 1 (2022): 634–42. https://doi.org/10.1609/aaai.v36i1.19943.

[8] Lewis, Patrick, Ethan Perez,

[9] Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, et al. “Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks.” arXiv.org, April 12, 2021. https://arxiv.org/abs/2005.11401.

[10] Ramzan, S., Iqbal, M. M., & Kalsum, T. (2022). Text-to-image generation using Deep Learning. IEEC 2022. https://doi.org/10.3390/engproc2022020016

[10] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang and Tatsunori B. Hashimoto. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca, 2023.

[11] Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language model with self generated instructions. arXiv preprint arXiv:2212.10560, 2022.

[12] R. E. Kalman and R. S. Bucy, “New results in linear filtering and prediction theory,” Trans. ASME Series D, J. Basic Engineering, pp. 95-108, 1961.

[13] Shayne Longpre, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V Le, Barret Zoph, Jason Wei, et al. The flan collection: Designing data and methods for effective instruction tuning. arXiv preprint arXiv:2301.13688, 2023.

[14] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. arXiv e-prints, 2019. URL https://arxiv.org/abs/1910.10683.

[15] Adam Roberts, Colin Raffel, and Noam Shazeer. How much knowledge can you pack into the parameters of a language model? arXiv e-prints, 2020. URL https://arxiv.org/abs/2002.08910.

[16] Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: Bm25 and beyond. Found. Trends Inf. Retr., 3(4):333–389, April 2009. ISSN 1554-0669. doi: 10.1561/1500000019. URL https://doi.org/10.1561/1500000019.

[17] Agrawal, A.; Lu, J.; Antol, S.; Mitchell, M.; Zitnick, C. L.; Batra, D.; and Parikh, D. 2015. TGIF-QA: Toward Spatio-Temporal Reasoning in Visual Question Answering. In ICCV. AllenAI2. 2018. ScienceParse. https://reurl.cc/e62LXL. Accessed: 2020-09-04.

[18] Tony Lacey, “Likelihood interpretation of Kalman filter”, tutorial lecture, April 2006.

[19] Marchesotti, L.; Perronnin, F.; Larlus, D.; and Csurka, G. 2011. VMSMO: Learning to Generate Multimodal Summary for Video-based News Articles. In ICCV. Microsoft. 2021a. Azure Cognitive Services. https://reurl.cc/Qjqe45. Accessed: 2020-09-04.

[20] Rush, A. M.; Chopra, S.; and Weston, J. 2015. A Neural Attention Model for Abstractive Sentence Summarization. In EMNLP. See, A.; Liu, P. J.; and Manning, C. D. 2017. Get To The Point: Summarization with Pointer-Generator Networks. In ACL.

[21] Deep reinforcement learning from human preferences. In Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9,2017, Long Beach, CA, USA, pages 4299–4307

[22] Tao Gui, Xiao Wang, Qi Zhang, Qin Liu, Yicheng Zou, Xin Zhou, Rui Zheng, Chong Zhang, Qinzhuo Wu, Jiacheng Ye, et al. 2021. Textflint: Unified multilingual robustness evaluation toolkit for natural language processing. arXiv preprint arXiv:2103.11441.

[23] Simran Arora, Avanika Narayan, Mayee F. Chen, Laurel Orr, Neel Guha, Kush Bhatia, Ines Chami, Frederic Sala, and Christopher R´e. Ask me anything: A simple strategy for prompting language models, 2022.

[24] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. Advances in neural information processing systems, 33:1877–1901, 2020.

[25] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. CoRR, abs/1810.04805, 2018

[26] Chenlin Meng, Robin Rombach, Ruiqi Gao, Diederik Kingma, Stefano Ermon, Jonathan Ho, and Tim Salimans. On distillation of guided diffusion models. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 14297–14306, 2023.

[27] Guillermo Ortiz-Jimenez, Alessandro Favero, and Pascal Frossard. Task arithmetic in the tangent space: Improved editing of pre-trained models. arXiv preprint arXiv:2305.12827, 2023.

[28] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Muller, JoePenna, and Robin Rombach. Sdxl: improving latent diffusion models for high-resolution image synthesis. arXiv preprint arXiv:2307.01952, 2023.

[29]DiederikP. Kingmaand MaxWelling. Auto-EncodingVariational Bayes. In 2nd International Conference on Learning Representations, ICLR, 2014

[30] Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. Diffwave: A versatile diffusion model for audio synthesis. In ICLR. OpenReview.net, 2021

[31] Hugging Face, StabalityAI, https://huggingface.co/stabilityai/sdxl-turbo

[32] https://tomthedesigner.com/how-long-does-it-take-to-make-a-presentation/