Team 10: DocuPresenter - Prompt to Slides

Final Report

Experiential Learning: Mini 2
Professor Ganesh Mani

Aniruddh Ingle
Sylvanne Braganza
Sherry Li
Hritik Bhansali
Cece Zhang

DECEMBER 9TH 2023

Table of Contents

1. Project Overview

Our project, DocuPresenter, aims to transform text-based content into engaging presentations with the help of Google's powerful PaLM2 API. By harnessing the capabilities of this cutting-edge natural language processing tool, we have streamlined the process of creating dynamic presentations from textual information. Whether it is converting lengthy reports into informative slides or transforming written content into compelling spoken presentations, our team has enhanced the efficiency and effectiveness of communication. With the PaLM2 API, we automated the structuring and summarizing of information, allowing users to effortlessly transform their ideas and data into engaging spoken presentations. We have also seamlessly integrated the capabilities of Google Translate API and Google Drive. This comprehensive approach ensures that users can effortlessly create dynamic presentations with multilingual support while managing their presentation documents efficiently through Google Drive. Moreover, our project offers customization options, allowing users to tailor presentations based on age, language, and topic preferences. Whether it's crafting presentations for different age groups, diverse languages, or specific topics, our tool provides a versatile and user-friendly solution. This project is a game-changer in simplifying the presentation creation process, making it an invaluable tool for professionals, educators, and anyone looking to communicate their ideas effectively.

2. Hypothesis

The effective transformation of text into engaging presentations can be achieved by leveraging Google's PaLM2 API and MakerSuite for prompt engineering. Through iterative fine-tuning of the model using data prompts, initially breaking text into bullet points and later generating Markdown-formatted scripts for multi-slide presentations, we aim to explore the impact of deterministic and random characteristics in the PaLM2 model. By controlling randomness through temperature settings during prompt generation, we anticipate that the refined model responses will exhibit enhanced creativity and coherence.

Furthermore, the integration of the Google Translate API for multilingual expansion will be a key aspect of our project. We hypothesize that incorporating the translation capabilities of Google

Translate into our presentation generation process will not only enable the transformation of text into multiple languages but also enhance the overall inclusivity and global reach of the presentations. The seamless integration of the Google Translate API alongside PaLM2 and MakerSuite is expected to provide a comprehensive solution for creating engaging presentations in a multilingual context.

Additionally, the seamless integration of the refined model responses into Python code within Google Colab, in conjunction with the PaLM API, is hypothesized to facilitate efficient and streamlined presentation creation. We expect that the connection of input and output PDF files to our Google Drive account, facilitated by Google APIs Explorer and the Google Drive API, will enhance accessibility and convenience in managing presentation-related documents. This comprehensive approach, combining PaLM2, MakerSuite, Google Translate API, and Google Drive integration, is anticipated to result in an innovative and efficient solution for text-to-presentation transformation with improved user experience, accessibility, and multilingual capabilities.

3. Methodology

3.1 Makersuite

In our project, we leveraged Google's MakerSuite for prompt engineering. Initially, we explored different prompt types, including text prompts, data prompts, and chat prompts. After experimenting with these options, we found that the data prompt was the most suitable choice for our project aims due to its ability to tailor the input and output structure according to our project's specific requirements (text to Markdown-formatted script).

Data prompts work by taking custom instructions provided by users. In our case, we asked it to generate a presentation in Markdown script format tailored to a specific age group. Then we give the model specific input and output examples to work with. These prompts give the model examples to replicate. Google calls these "few-shot prompts." They served as training data for the model. Over time, we fine-tuned these examples based on the specific content structure and style we wanted for our presentations. This iterative process allowed the model to learn and adapt to our requirements. We proceeded step by step.

Our initial goal was to use this data prompt to efficiently break blocks of text into bullet points, simplifying the content structuring process. Once we achieved this, we took it a step further by instructing the data prompt to format the extracted bullet points as a Markdown script for a two-slide presentation, consisting of one title slide and one content slide. Finally, we tailored our examples to generate multi-slide presentation Markdown outputs, providing even greater flexibility and automation in the presentation creation process.

Our input texts were initially taken from Wikipedia articles; our outputs were initially written based on a template HTML Markdown code (Appendix A) that we manually filled in. In MakerSuite, we were then able to have the model use our examples (Appendix B) to generate more examples.

We also experimented with the underlying LLM at this stage. LLM responses exhibit both deterministic and random characteristics. When you input a prompt to an LLM, it generates a probability distribution over potential tokens (words) likely to appear next. This initial stage is entirely deterministic; the LLM consistently produces the same distribution when given the same prompt. However, in the subsequent stage, the LLM transforms these distributions into actual text responses using various decoding methods. A straightforward decoding approach might select the most probable token at each step, ensuring determinism. Alternatively, you can opt for a response generation method that involves random sampling from the model's output distribution, introducing a degree of randomness. This stochastic behavior can be controlled by adjusting the temperature setting. A temperature of 0 results in deterministic token selection, while higher temperatures introduce more randomness, yielding unexpected and surprising model responses. (LLM Concepts Guide, 2023)

In our prompt generation stage, we changed this temperature setting to achieve a broad range of example prompts that still formatted the output as our desired Markdown code script. MakerSuite then allowed us to add the newly generated examples that we approved as more examples to train the model.

Once we were happy with the model's responses to our test inputs, we easily exported it to Python code (Appendix C) in Google Colab, and called the same model using the PaLM API.

3.2 Document Search with Embedding

This technique allows for searching documents by comparing their embedded representations, or vectors of numbers that capture the semantic meaning of text. There are several advantages to using document search with embedded text over keyword-based, traditional search methods for our project. The main reason is that it is more robust to synonyms, misspellings, which might crop up in our users' queries or in the conversion from the input PDF document(s) to string(s). Another advantage is that it can be used to search for documents in a variety of languages, since it does just depend on semantic meanings.

The steps are as follows. First, we generated embedded representations for all of the text in the document(s). This was done using a pre-trained embedding model, a few of which are provided by Google. Next, we generated an embedded representation of the query text. This was done using the same embedding model as used in the first step. Then, we can compare the embedded representation of the query text to the embedded representations of the input document(s). This was done using similarity metrics, such as distance determinations. Finally, we returned the results with the highest scores.

We based our code on the example provided by Google here:
https://developers.generativeai.google/examples/doc_search_emb.

This code allowed us to use the PaLM API to create embeddings so that we could perform document search. In our code, we can generate embeddings, build an embeddings database, and then document search with a Q&A system. The embedded representations are compared using the dot product of the vectors, ranging from -1 to 1.

We then incorporated our engineered prompts into our document search with embeddings code.

3.3 Google API

Connecting our input and output files to our Google Drive account has significantly enhanced the ease of use and provided seamless access to input and output documents. Leveraging the capabilities of the Google Drive API through Google APIs Explorer, our system now efficiently interacts with Google Drive resources. This integration lets our model seamlessly retrieve and store files, streamlining the entire process and ensuring that our users can effortlessly manage and access their documents and presentations directly through their Google Drive accounts.

3.4 Google Translate

In order to create a more inclusive and globally accessible platform, we have seamlessly integrated the Google Translate API to facilitate multilingual support within our system. Despite our model being initially designed in English, this integration enables us to generate presentations in any language supported by Google Translate. This means that our users can now effortlessly transform content into a diverse array of languages, transcending the language limitations of the model's native proficiency. Regardless of the language of the original input text, our users can leverage the power of the Google Translate API to produce presentations that cater to a global audience, fostering inclusivity and accessibility.

3.5 Expanding Inputs

To broaden the reach and adaptability of our project, we also decided to modify our project for input expansion. Our testing had been confined to the searchability of queries within PDF documents. However, recognizing the need for versatility, we expanded our model's capabilities to include Word documents. This was geared towards enhancing the adaptability of our query system, enabling users to explore a wider array of document types and lengths.

3.6 Customization

We allow our users to customize their presentation in three ways (Appendix G).

3.6.1 Age

The aspect of audience appropriateness plays a pivotal role in tailoring content to different age groups.

For example, we specifically compared two presentations, one generated for 13-year-olds and one generated for 23-year-olds.

For 13-year-olds, our focus is on presenting information in a simplified and engaging language, ensuring easy comprehension. The content introduces basic concepts, employing everyday examples of Artificial Intelligence (AI) and Machine Learning (ML) to spark interest and familiarity. On the other hand, for 23-year-olds, the tone and complexity shift to meet the expectations of a more educated audience. The language becomes sophisticated and technical, offering detailed discussions on AI and ML, including specific algorithms.

In terms of the complexity of concepts, the content is meticulously adjusted for each age group. For 13-year-olds, the emphasis is on providing a basic-level introduction to concepts, laying the groundwork for future understanding. In contrast, 23-year-olds encounter a more intricate exploration, delving into the nuances of AI and ML, offering a comprehensive coverage that caters to an advanced understanding of the subject.

The application and examples employed further differentiate the content. For 13-year-olds, the focus is on general, everyday examples of AI and ML, making the concepts relatable and tangible. In contrast, 23-year-olds are presented with in-depth industry applications and real-world impact scenarios, connecting theoretical knowledge to practical implications.

This tailored approach ensures that the depth of content aligns with the audience's age, fostering engagement and understanding while providing a scaffold for continuous learning and exploration.

3.6.2 Language

Ensuring adaptability to diverse audiences, it is imperative that the presenter has the capability to deliver their presentation in various languages. By integrating the Google Translate API, Docupresenter facilitates the generation and summarization of content in any language, covering a broad spectrum of subjects.

For instance, we replicated the Artificial Intelligence example discussed earlier, creating a presentation in Spanish. The incorporation of multilingual functionality positions Docupresenter as a universally applicable tool, transcending borders and appealing to a global and diversified audience. Further language conversions, including Hindi and German, are detailed in the Appendix.

3.6.3 Topic

An uploaded article available for conversion may encompass a range of topics, even within a specific subject. Consider an article on the American Civil War, covering its causes, reasons, effects, and more.

Docupresenter empowers users to tailor their presentations by specifying keywords or topics of interest, allowing for the extraction of relevant information from the article.
This customization results in a more accurate and targeted presentation that aligns with the user's preferences.

4. Results

DocuPresenter employs Google Generative AI for both text embedding and generation. It leverages the PyPDF2 library to extract text from PDF files.

After installing necessary packages and importing libraries for set up, DocuPresenter first mounts Google Drive to obtain access to PDF files in drive folders. Then it extracts text from PDF files in a specified folder, cleans the text by removing newline characters, and stores the processed text in a Pandas DataFrame. After file retrieval, it utilizes Google Generative AI to generate embeddings for the text extracted from PDFs and then adds the generated embeddings to a DataFrame.

DocuPresenter then asks users to specify a topic. Then it defines a function to find the most relevant passage across all documents in the specified folder. Specifically, It queries the documents to find the best passage based on the specified topic. After querying the documents, DocuPresenter generates a presentation using the specified prompt, a text generation model, and parameters including temperature. Then it converts the generated Markdown content to a PDF and saves it in a folder on Google Drive.

We have tested DocuPresenter’s performance on books and other resources across various disciplines, including Economics, Investment, History, Machine Learning, Organizational Behavior, and Architecture.

How to Use

We have included a ReadMe file in our submission (Appendix F).

Briefly, a user can upload a PDF or Word document into a folder in their Google Drive. They can then adjust the query to specify topic, language, and age or education level. Finally, they run our python script to generate a PDF presentation of slides, which will be saved back into their Google Drive.

5. Evaluation

After completing our project, we ran it through an evaluation process in order to see where improvements might be made. To comprehensively assess the efficacy of our model, we systematically generated 50 distinct presentations across various disciplines, for various ages, and in various languages. We then evaluated each presentation along two dimensions. The first dimension gauged the accuracy of the results, scrutinizing whether the content adhered to the information provided in the input documents or, conversely, introduced extraneous information. This evaluation criterion aimed to ensure precision and reliability in the generated content. The second dimension delved into the presentation quality, considering factors such as the coherence of content summaries and the visual appeal of the slides. We scrutinized the organization of information into bullet points, the introduction of new slides for each heading, and the implementation of stylistic elements like bolding and italics. Thus, each presentation received a comprehensive score out of five, with five being the best, providing insights into the strengths and areas for improvement of our project.

5.1 Results of Evaluation

Based on our evaluation (Appendix E), out of 50 generated presentations, the average score for Content was 4.28 out of 5 (85.60%). The average score for Formatting was 3.92 out of 5 (78.40%).

The biggest issues we found in regards to formatting were insufficient use of stylistic elements, poorly separated slides with content split across slides, and extraneous symbols.

The biggest issues we found in regards to content were difficulties in nuance and clear, summarized content in languages other than English and occasionally content that was not derived from our input sources.

5.2 Improvements Based on Evaluation

We have fine-tuned the formatting of the slides. Now, the slides are separated based on each header, creating a clearer and more structured presentation. Furthermore, we have removed extraneous symbols, such as the pound symbol, from the output text, contributing to a cleaner visual. These targeted improvements address specific pain points identified during evaluation. but also underscore our commitment to continual enhancement, ensuring that our model evolves in tandem with user needs and industry standards.

Another improvement in our code was related to the extraction of text from PDF documents. For some reason, possibly due to the encoding of the PDF files, occasionally the extracted text had no spacing. As a result, when the LLM processed the text, it interpreted it as a single, uninterrupted string, rendering it virtually unusable for further analysis. To address this issue, we implemented a line of code that ensured the insertion of spaces between words, thus enabling the LLM to interpret and analyze the text correctly, enhancing the overall functionality of our system.

6. Challenges

We encountered 3 main challenges in DocuPresenter's development:

1. Undiversified sources: In the first half of this mini, DocuPresenter was only able to take in PDF documents. Now it is capable of reading .docx files from Google Drive too. However, the byte-limit for .docx documents is stricter than for .pdf documents. Therefore, we incorporated in our codes a query to convert .docx files into .pdf files to bypass the restriction.
2. The embedding system's limitation: The embedding system caps data processing at 10,000 bytes or approximately 5,000 words. To overcome the error this generated in our code, we integrated a code line processing text stored in the 'cleaned_text' variable. Initially, the code encoded the text into bytes using UTF-8 encoding, a standard practice for byte representation. Subsequently, the code sliced the first 9900 bytes from the encoded sequence, fitting the size constraints. However, this is only a temporary solution. We hope to unlock DocuPresenter's potential without this restriction in the future.
3. Multilingual expansion attempt: Our testing was predominantly focused on prompts in the English language, and we attempted to extend our testing to encompass languages beyond English. We experimented with PaLM2, but the LLM only generates outputs in English. Therefore, the method we adopted was to incorporate a translation API into our code.

However, this is not the most ideal solution because it loses out on the nuances created by the LLM model.

7. Future Improvements

7.1 Full Potential of Document Processing

The current data processing limitation of 10,000 bytes. Looking ahead, our aim is to unleash the full potential of DocuPresenter by addressing and eliminating this restriction in the future. We are committed to refining our system to enable seamless data processing beyond the current limitations.

7.2 Methods for User-Centric Improvement

We can evaluate if there are ways to optimize the representation of text data without losing essential information. This might involve compressing the data or removing unnecessary elements.

7.3 Methods for Technological Improvements

1. Data Pagination: Introduce a pagination system for processing large amounts of text.
2. Algorithmic Improvements: Explore if there are more efficient algorithms or data structures that can be employed for text processing.
3. File Format Conversion: We have discovered that using PDF as an input provides the model with more content compared to DOCX. To enhance the information available for the model, we plan to explore converting various other file formats, such as web pages, into PDFs that can be used by our model. This will enable a broader range of document content to be processed by the model.

7.4 Optimizing Multilingual Data Processing

Up until now, our approach involves generating slides and subsequently translating their content into different languages using the Translate API. However, our ideal scenario is to integrate the translation process during data retrieval. The goal is to translate all documents at the time of data ingestion, allowing us to then extract the appropriate information based on the translated content.

1. Parallel Translation Integration: Implement a parallel translation integration during the data retrieval phase. As documents are ingested, trigger simultaneous translations in multiple languages using Google Translate.
2. Context-Aware Translation Services: Employ translation services that are context-aware and tailored to specific domains. The services can offer more accurate translations by considering the industry-specific terminology and nuances present in the documents.

7.5 Introducing a Streamlined UI

We anticipate providing users with a user interface (UI) that allows for files/pictures upload and download, along with the input of topics and language preferences. Currently, there is a need to modify queries within the model, but the introduction of a UI is expected to streamline the overall process, making it faster and more convenient.

Additionally, we aim to enhance user experience by incorporating a feature that enables template selection based on presentation topics.

1. File Management System: Develop a user-friendly file management system within the UI that allows users to easily upload and download files.
2. Template Selection Feature: Introduce a template selection feature in the UI that allows users to choose presentation templates.
3. User Profile and Preferences: Create user profiles within the UI where users can save their preferences, including preferred topics and language settings.

7.6 Elevating Presentation Quality

Currently, our system generates words instead of images, and the header of each slide lacks the desired boldness and center alignment. Our goal is to enhance the visual appeal and overall quality of generated presentations.

Additionally, we aim to enhance user experience by incorporating a feature that enables template selection based on presentation topics.

1. Template Selection Feature: Introduce a template selection feature in the UI that allows users to choose presentation templates.
2. Image & Audio Generation Integration: Explore image generation libraries or services to enable the generation of more visually engaging presentations.

3. Style Enhancement for Headers: Adjust the styling parameters during the presentation creation process to ensure headers on each slide are bold and centrally aligned.

8. Conclusion

Our project aims to transform text into dynamic presentations using Google's PaLM2 API and integrating MakerSuite. We fine-tune the model through iterative prompt engineering for creative and coherent outputs.

We use Google Translate API for multilingual expansion, enabling presentations in various languages. The streamlined approach involves Google Generative AI for text embedding, Python code in Google Colab, and seamless integration with Google Drive for efficient presentation management. DocuPresenter, leveraging Google Generative AI, extracts and processes text from diverse documents. There are challenges, such as source diversification and data processing limitations, which have led to solutions like file format conversion for broader content processing.

Future improvements include addressing data processing limitations and optimizing text data representation. We also aim to explore image generation, enhance header styling, and introduce a template selection feature for visually appealing presentations.

In summary, our project offers an innovative solution, combining advanced AI technologies and user-centric features to streamline presentation creation for professionals and educators.

Team Contributions

- Document Search with Embeddings/DocuPresenter, Query (Code) and Google API/Retrieving files from Data (Code), Google Translate API (Code) - ***Aniruddh***
- Converting MD to PDF, Saving to Drive (Code) - ***Hritik, Aniruddh***
- DOCX to PDF (Code), Document match alert - ***Sherry, Cece, Aniruddh***
- MakerSuite (Code) and Formatting (Code), Evaluation Metrics - ***Sylvanne, Sherry***
- ReadMe - ***Sherry, Sylvanne***
- Overview, Hypothesis, MakerSuite, Document Search, Google API, Google Translate, Input Expansion, How to, Evaluation (Report) - ***Sylvanne***
- Future Directions, Conclusion (Report) - ***Sherry***
- Appendix, Citations, Results, Challenges, Future Directions (Report) - ***Cece***
- Customization (Report) - ***Aniruddh, Hritik, Sylvanne***
- Report Formatting (Report) - ***Aniruddh, Sherry, Sylvanne***
- MakerSuite Prompt Engineering - ***Sherry, Sylvanne, Hritik, Cece***
- Presentation Slides - ***Sherry, Cece, Sylvanne, Hritik, Aniruddh***
- Presenters - ***Hritik, Aniruddh***
- Video - ***Aniruddh***

Appendix

Appendix A

Input Examples:
https://drive.google.com/drive/folders/11T8hEyjsjK-eeJAn5rJ6SJ6Y32T16bRb?usp=sharing

Output Examples:
https://drive.google.com/drive/folders/153eEK6Vqr9VMtYlyL_8jhBVxnBQ4t-Ym?usp=drive_link

Appendix B

MarkDown Output Template Code:

```
import markdown

# Title and subtitle for the first slide
  title = ""
subtitle = ""

# Slide content for the second slide
slide_title = ""
slide_content = ""

# Slide content for the third slide
slide_title2 = ""
slide_heading2 = ""
slide_content2 = ""

# Slide content for the fourth slide
slide_title3 = ""
slide_heading3 = ""
slide_content3 = "l"

# Slide content for the fifth slide slide_title4 = "" slide_heading4 = ""
slide_content4 = ""

# Slide content for the sixth slide
slide_title5 = ""
slide_heading5 = ""
slide_content5 = ""
```

```
# Create the markdown content
#title: {title}
#subtitle: {subtitle}
 markdown_content = f"""---

# {title}
## {subtitle}
---
# {slide_title_1}
## {slide_heading_1}

{slide_content_1}
"""
---
# {slide_title_2}
## {slide_heading_2}

{slide_content_2}
"""
---
# {slide_title_3}
## {slide_heading_3}

{slide_content_3}
"""



# Convert the markdown content to HTML
 html_content = markdown.markdown(markdown_content)

# Print or save the markdown content
 print(markdown_content)

with open("output.html", "w") as f:
    f.write(html_content)
```

Appendix C

Prompt Examples:

	Input	Output

1	The Renaissance is a period in history and a cultural movement marking the transition from the Middle Ages to modernity, covering the 15th and 16th centuries and characterized by an effort to revive and surpass the ideas and achievements of classical antiquity; it occurred after the crisis of the Late Middle Ages and was associated with great social change in most fields and disciplines, including art, architecture, politics, literature, exploration, and science. In addition to the standard periodization, proponents of a "long Renaissance" may put its beginning in the 14th century and its end in the 17th century. The traditional view focuses more on the Renaissance's early modern aspects and argues that it was a break from the past, but many historians today focus more on its medieval aspects and argue that it was an extension of the Middle Ages. The beginnings of the period—the early Renaissance of the 15th century and the Italian Proto-Renaissance from around 1250 or 1300—overlap considerably with the Late Middle Ages, conventionally dated to, and the Middle Ages themselves were a long period filled with gradual changes, like the modern age; as a transitional period between both, the Renaissance has close similarities to both, especially the late and early sub-periods of either. The Renaissance's intellectual basis was its version of humanism, derived from the concept of Roman humanitas and the rediscovery of classical Greek philosophy. Early examples were the development of perspective in oil painting and the revived knowledge of how to make concrete. Although the invention of metal movable type sped the dissemination of ideas from the later 15th century, the changes of the Renaissance were not uniform across Europe: the first traces appear in Italy as early as the late 13th century, in particular	import markdown
		
		# Title and subtitle for the first slide title = "The Renaissance: A Cultural Revolution"
		subtitle = "Exploring the Origins, Characteristics, and Impact"
		
		# Slide content for the second slide slide_title = "Introduction"
		slide_heading = "Defining the Renaissance" slide_content = "Main Ideas.\n\n- Transition from the Middle Ages to modernity\n- 15th and 16th centuries\n- Revival of classical antiquity\n- Variations in defining the Renaissance\n- The concept of a long Renaissance"
		
		# Slide content for the third slide
		slide_title2 = "Renaissance Features" slide_heading2 = "Characteristics of the Renaissance"
		slide_content2 = "The main characteristics are:\n\n- Humanism and classical Greek philosophy\n- Perspective in painting and concrete knowledge\n- Resurgence of learning and natural reality in art\n
		Diplomacy and observation in politics and science\n- Intellectual and social scientific progress"
		
		# Slide content for the fourth slide
		slide_title3 = "Renaissance in Italy" slide_heading3 = "Origins and Centers in Italy"
		slide_content3 = "In Italy:\n\n- Florence as the starting point\n- Political structure and Medici patronage\n- Venice, Genoa, Milan,
		Rome, and Naples\n- The role of the Medici family\n- Greek scholars' migration after Constantinople's fall"
		
		# Slide content for the fifth slide

	with the writings of Dante and the paintings of Giotto. As a cultural movement, the Renaissance encompassed innovative flowering of Latin and vernacular literatures, beginning with the 14th-century resurgence of learning based on classical sources, which contemporaries credited to Petrarch; the development of linear perspective and other techniques of rendering a more natural reality in painting; and gradual but widespread educational reform. In politics, the Renaissance contributed to the development of the customs and
	conventions of diplomacy, and in science to an increased reliance on observation and inductive reasoning. Although the
	Renaissance saw revolutions in many intellectual and social scientific pursuits, as well as the introduction of modern banking and the field of accounting, it is perhaps best known for its artistic developments and the contributions of such polymaths as Leonardo da Vinci and Michelangelo, who inspired the term "Renaissance man". The Renaissance began in Florence, one of the many states of Italy. Various theories have been proposed to account for its origins and characteristics, focusing on a variety of factors, including Florence's social and civic peculiarities at the time: its political
	structure, the patronage of its dominant family, the Medici, and the migration of Greek scholars and their texts to Italy following the fall of Constantinople to the Ottoman Turks. Other major centers were Venice, Genoa, Milan, Rome during the Renaissance Papacy, and Naples. From Italy, the Renaissance spread throughout Europe and also to American, African, and Asian territories ruled by the European colonial powers of the time or where Christian missionaries were active.
		slide_title4 = "Spread of the Renaissance" slide_heading4 = "Dissemination Across Europe and Beyond"
		slide_content4 = "\n\n- How the
		Renaissance spread in Europe\n- Its influence on American territories\n
		Renaissance's impact on African regions\n Role of Christian missionaries\n- The global reach of the Renaissance"
		
		# Slide content for the sixth slide
		slide_title5 = "Legacy of the Renaissance" slide_heading5 = "Enduring Impact" slide_content5 = "The main impacts:\n\n Contributions to art and artists like da Vinci and Michelangelo\n- Intellectual and scientific advancements\n- Influence on modern diplomacy\n- Economic and financial changes\n- The concept of the Renaissance Man and its significance"
		
		# Create the markdown content
		#title: {title}
		#subtitle: {subtitle}
		markdown_content = f"""---
		
		# {title}
		## {subtitle}
		
		—
		
		# {slide_title}
		## {slide_heading}
		
		{slide_content}
		
		--- 	
		
		# {slide_title2}
		## {slide_heading2}
		
		{slide_content2}
		
		--- 	
		
		# {slide_title3}

		## {slide_heading3}
		
		{slide_content3}
		
		--- 	
		
		# {slide_title4}
		## {slide_heading4}
		
		{slide_content4}
		
		--- 	
		
		# {slide_title5}
		## {slide_heading5}
		
		{slide_content5}
		
		"""
		
		# Convert the markdown content to HTML (optional) html_content =
		markdown.markdown(markdown_content)
		
		# Print or save the markdown content
		print(markdown_content)
		
		with open("output.html", "w") as f:
		f.write(html_content)

	World War II (WWII or WW2) or the Second World War was a global conflict that lasted from 1939 to 1945. The vast majority of the world's countries, including all the great powers, fought as part of two opposing military alliances: the Allies and the Axis. Many participants threw their economic, industrial, and scientific capabilities behind this total war, blurring the distinction between civilian and military resources. Aircraft played a major role, enabling the strategic bombing of population centres and delivery of the only two nuclear weapons ever used in war. It was by far the deadliest conflict in history, resulting in 70 to 85 million fatalities, mostly civilians. Millions died due to genocides, including the Holocaust, as well as starvation,	import markdown
		
		# Title and subtitle for the first slide title = "World War II: The Global Conflict of 1939-1945"
		subtitle = "Causes, Alliances, and Impact on Humanity"
		
		# Slide content for the second slide slide_title = "World War II: The Global Conflict"
		slide_heading = "Key Aspects of World War II"
		slide_content = "Here are the Key
		Aspects.\n\n- Duration: 1939 to 1945\n- Two Major Alliances: Allies and Axis\n
		Economic, Industrial, and Scientific

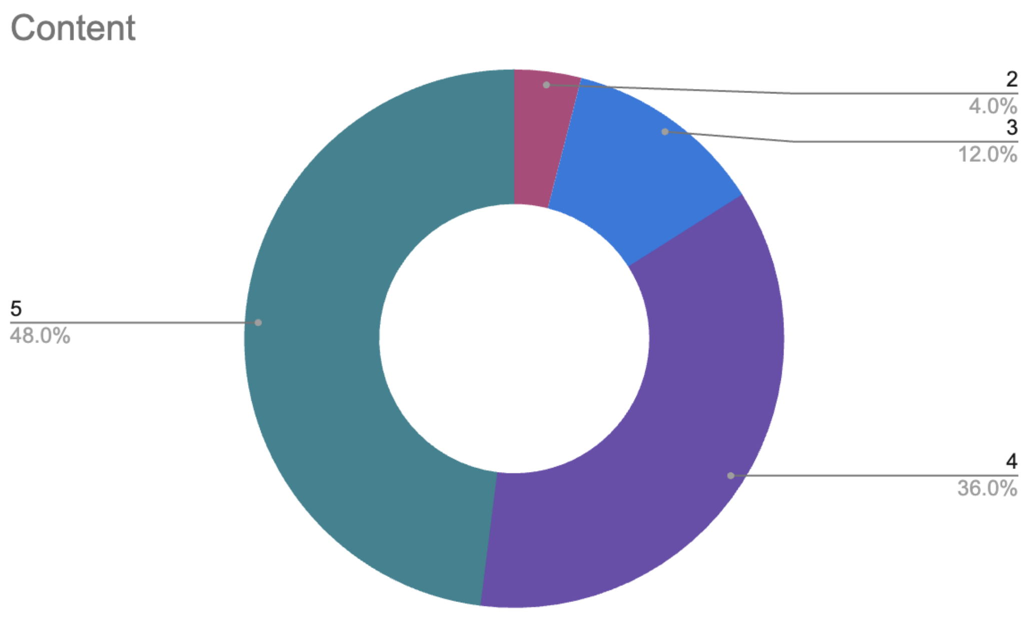
Appendix D

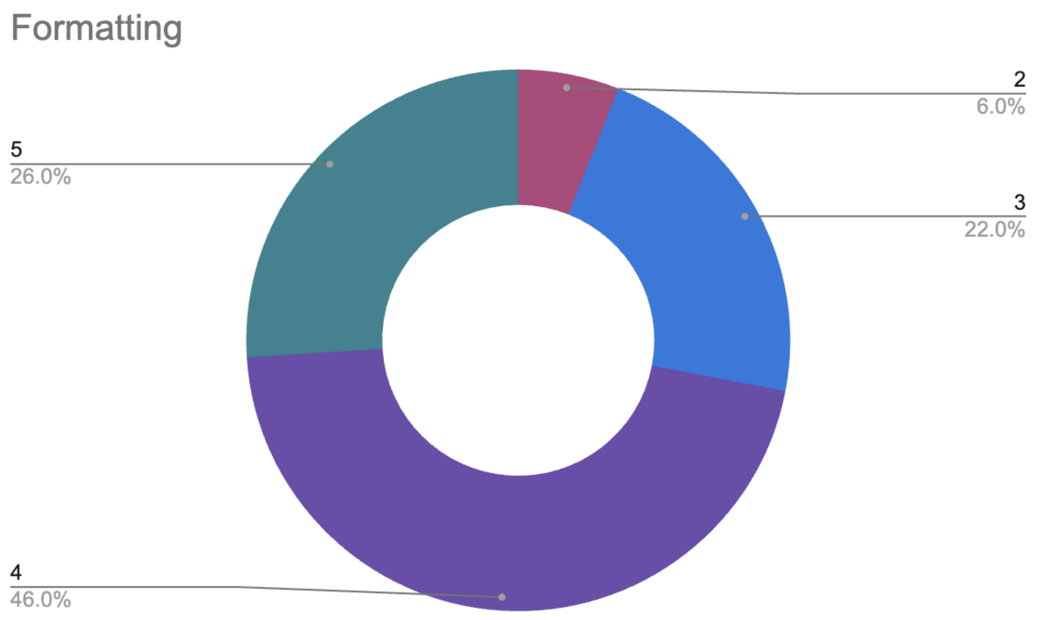
Engineered Prompt Python Code:

Copy of makersuite_text_prompt.ipynb

Appendix E

Evaluation Responses:



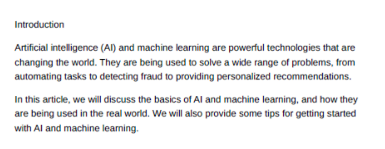


Appendix F

ReadMe

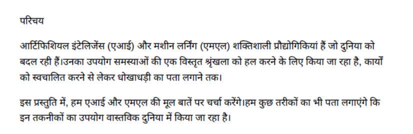
Appendix G

Multilingual Comparison (English, Hindi, German excerpts from generated presentations):


Introduction

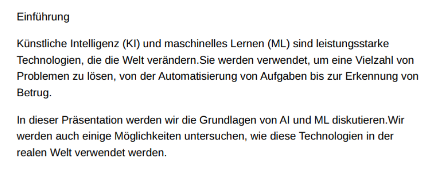
Artificial intelligence (AI) and machine learning are powerful technologies that are changing the world. They are being used to solve a wide range of problems, from automating tasks to detecting fraud to providing personalized recommendations.

In this article, we will discuss the basics of AI and machine learning, and how they are being used in the real world. We will also provide some tips for getting started with AI and machine learning.


परिचय

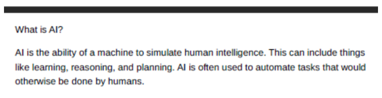
आर्टिफिशियल इंटेलिजेंस (एआई) और मशीन लर्निंग (एमएल) शक्तिशाली प्रौद्योगिकियां हैं जो दुनिया को बदल रही हैं।उनका उपयोग समस्याओं की एक विस्तृत श्रृंखला को हल करने के लिए किया जा रहा है, कार्यों को स्वचालित करने से लेकर धोखाधड़ी का पता लगाने तक।

इस प्रस्तुति में, हम एआई और एमएल की मूल बातें पर चर्चा करेंगे।हम कुछ तरीकों का भी पता लगाएंगे कि इन तकनीकों का उपयोग वास्तविक दुनिया में किया जा रहा है।

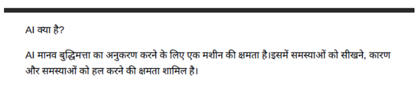

Einführung

Künstliche Intelligenz (KI) und maschinelles Lernen (ML) sind leistungsstarke Technologien, die die Welt verändern.Sie werden verwendet, um eine Vielzahl von Problemen zu lösen, von der Automatisierung von Aufgaben bis zur Erkennung von Betrug.

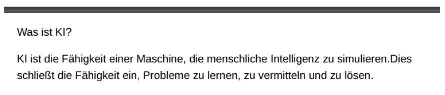
In dieser Präsentation werden wir die Grundlagen von AI und ML diskutieren.Wir werden auch einige Möglichkeiten untersuchen, wie diese Technologien in der realen Welt verwendet werden.


What is AI?

AI is the ability of a machine to simulate human intelligence. This can include things like learning, reasoning, and planning. AI is often used to automate tasks that would otherwise be done by humans.

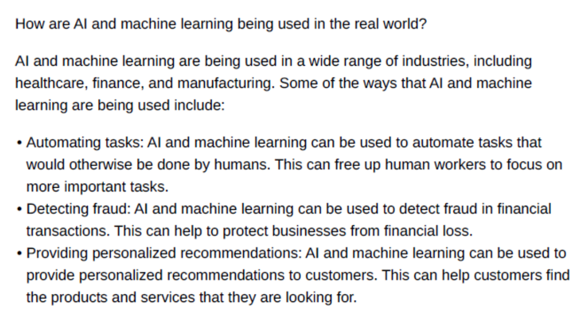

AI क्या है?

AI मानव बुद्धिमत्ता का अनुकरण करने के लिए एक मशीन की क्षमता है।इसमें समस्याओं को सीखने, कारण और समस्याओं को हल करने की क्षमता शामिल है।


Was ist KI?

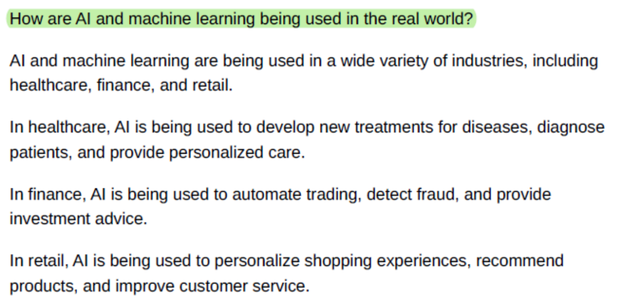
KI ist die Fähigkeit einer Maschine, die menschliche Intelligenz zu simulieren.Dies schließt die Fähigkeit ein, Probleme zu lernen, zu vermitteln und zu lösen.

Age Comparison (13, 23):


How are AI and machine learning being used in the real world?

AI and machine learning are being used in a wide range of industries, including healthcare, finance, and manufacturing. Some of the ways that AI and machine learning are being used include:

- Automating tasks: AI and machine learning can be used to automate tasks that would otherwise be done by humans. This can free up human workers to focus on more important tasks.
- Detecting fraud: AI and machine learning can be used to detect fraud in financial transactions. This can help to protect businesses from financial loss.
- Providing personalized recommendations: AI and machine learning can be used to provide personalized recommendations to customers. This can help customers find the products and services that they are looking for.

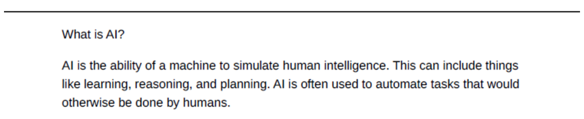

How are AI and machine learning being used in the real world?

AI and machine learning are being used in a wide variety of industries, including healthcare, finance, and retail.

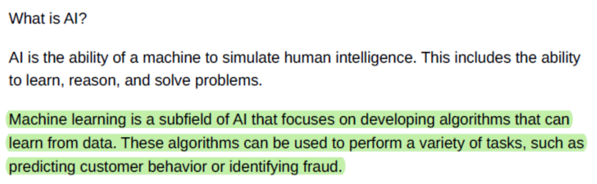
In healthcare, AI is being used to develop new treatments for diseases, diagnose patients, and provide personalized care.

In finance, AI is being used to automate trading, detect fraud, and provide investment advice.

In retail, AI is being used to personalize shopping experiences, recommend products, and improve customer service.


What is AI?

AI is the ability of a machine to simulate human intelligence. This can include things like learning, reasoning, and planning. AI is often used to automate tasks that would otherwise be done by humans.


What is AI?

AI is the ability of a machine to simulate human intelligence. This includes the ability to learn, reason, and solve problems.

Machine learning is a subfield of AI that focuses on developing algorithms that can learn from data. These algorithms can be used to perform a variety of tasks, such as predicting customer behavior or identifying fraud.

Citations

“Document Search with Embeddings : Palm API : Generative AI for Developers.” *Generative AI for Developers*, 1 Sept. 2023, developers.generativeai.google/examples/doc_search_emb.

“LLM Concepts Guide : Palm API : Generative AI for Developers.” *Generative AI for Developers*, 27 Sept. 2023, developers.generativeai.google/guide/concepts#prompt-types.