Multilingual Artifacts: Bridging Cultures through Natural Language Processing

Group Members

Hamza Khalid Baig
Hardik Kapadia
Ishika Aggarwal
Karthik Sivakumar
Shivam Dhiman

Abstract:

In an increasingly interconnected world, the preservation and dissemination of cultural heritage hold paramount importance. This project endeavours to bridge linguistic and cultural divides through the development of a Multimodal Language Model (LLM) tailored for museum artifact inquiries. The model facilitates user interaction in their native language, Tamil, enabling seamless communication and access to artifact information. Leveraging data scraping, question collection, translation, finetuning, and deployment, this endeavour showcases the fusion of cutting-edge Natural Language Processing (NLP) techniques to enhance user experience and accessibility to cultural artifacts.

1. Introduction:

Cultural artifacts encapsulate the essence of civilizations, serving as windows into the past and conduits for intercultural dialogue. However, language barriers often impede the widespread understanding and appreciation of these treasures. The museum selected for this Pilot phase of implementation was **National Museum, New Delhi.** This report presents a comprehensive methodology for developing a Multimodal Language Model (LLM) aimed at transcending linguistic barriers and fostering cross-cultural engagement.

2. Data Scraping:

The foundation of the project lies in the meticulous collection of artifact data. The data scraping process began by importing essential libraries. Selenium, the key player, facilitates control over a web browser for programmatic interaction. Pandas comes into the picture later, for data manipulation and storage.

Next, the script targeted the National Museum of India's collections webpage. The starting URL was defined, along with an unused class identifier, possibly intended for pagination. A clever technique then generated a list of URLs encompassing all artifact collection pages (pages 6 to 16).

With the URLs in hand, a Chrome WebDriver instance was initiated, essentially a virtual Chrome browser controlled by the script. The core logic then unfolded within a loop iterating through each collection page URL. On each page, the script meticulously identified all hyperlinks possessing the class name "modal-pop" using XPath, a powerful method for pinpointing specific elements on a webpage. These links likely corresponded to individual artifact details.

Another loop delved deeper, examining each extracted artifact link. Within a try-except block to manage potential errors, the script retrieved the title attribute (likely containing the artifact name) and the image source URL using XPath navigation. This extracted data, a testament to the script's success, was then printed for debugging purposes and subsequently appended to a list for later storage.

Finally, the script determined the total number of scraped artifacts and transformed the collected data into a structured format using pandas. This valuable information was then permanently preserved by saving it as a CSV file named "data.csv".

3. Question Collection:

The process of question collection in this project is pivotal for fostering meaningful interactions between users and the Multimodal Language Model (LLM), ensuring that the model can provide informative responses tailored to the artifacts in question. This section outlines the methodology employed for generating artifact-related questions and the role of Generative Pre-trained Transformer (GPT) in this process.

Before delving into question generation, the artifact data obtained through web scraping underwent meticulous preprocessing. This involved cleaning the data, extracting relevant metadata, and organizing it into a structured format. By standardizing the data, we ensured consistency and coherence in subsequent processing steps.

Generative Pre-trained Transformer (GPT), a state-of-the-art language model renowned for its ability to generate human-like text, played a central role in question generation. Leveraging the power of GPT, we aimed to craft questions that not only captured the essence of artifact descriptions but also stimulated informative responses from the model.

The question formulation process involved several key steps:

a. Understanding Artifact Descriptions:

The first step entailed comprehensively understanding the artifact descriptions derived from the metadata. This involved identifying key attributes, historical contexts, and unique features associated with each artifact.

b. Generating Diverse Question Types:

To ensure a holistic understanding of the artifacts, we crafted questions encompassing various aspects such as historical significance, cultural context, material composition, and artistic techniques. By diversifying question types, we aimed to elicit comprehensive responses from the model, enriching the user experience.

c. Tailoring Questions for Informative Responses:

Each question was carefully crafted to elicit informative responses that provide meaningful insights into the artifact. This involved formulating questions that prompt detailed explanations, descriptions, or interpretations, thereby enhancing the depth and richness of the interaction.

d. Iterative Refinement:

The question formulation process was iterative, involving continuous refinement and optimization based on feedback and evaluation. This iterative approach ensured that the questions evolved to effectively capture the nuances of artifact descriptions and cater to user inquiries comprehensively.

4. Translation:

Recognizing the significance of linguistic diversity, machine translation techniques were employed to bridge the gap between users' native language (Tamil) and English, the language of model training. GPT facilitated the generation of answers, subsequently translated back into Tamil to ensure user comprehension. This step mitigated language barriers, democratizing access to artifact information irrespective of linguistic background.

5. Finetuning:

The code snippet sets up a robust framework for training a language model, specifically LLaVa, integrating various libraries and techniques to optimize efficiency and performance. It begins by leveraging libraries like transformers, peft, bitsandbytes, and trl, each offering specialized functionalities crucial for model training and optimization. These libraries provide tools for working with pre-trained transformer models, fine-tuning for parameter efficiency, model quantization, and selective fine-tuning.

Key components are then assembled, including PyTorch for deep learning operations and classes like AutoTokenizer, TrainingArguments, and LlavaForConditionalGeneration from the transformers library. These components handle tasks such as loading pre-trained models, defining hyperparameters, and configuring the model for training. Notably, the incorporation of BitsAndBytesConfig and SFTTrainer demonstrates a focus on model optimization techniques like quantization and selective fine-tuning.

6. Deployment:

The conversation structure is defined through LLAVA_CHAT_TEMPLATE, ensuring a consistent format for user and assistant interactions. Text and image processing tools are prepared using AutoTokenizer and AutoProcessor, facilitating the conversion of text and image inputs into formats understandable by the model. The LLavaDataCollator class further streamlines data preparation by organizing training examples into batches and processing both text and image data accordingly.

Training data is fetched and split into training and evaluation sets using the datasets library, while hyperparameters for the training regimen are defined using TrainingArguments. The inclusion of LoraConfig for PEFT optimization highlights the focus on efficiency, with parameters tailored for parameter-efficient fine-tuning. Additionally, authentication with the Hugging Face Hub enables potential deployment and sharing of trained models.

The Trainer initialization encapsulates the entire training process, specifying the model, training arguments, datasets, optimization configuration, and data collation strategy. This comprehensive setup ensures that the LLaVa model is trained efficiently, learning from conversation examples to generate human-quality text responses. Overall, the code snippet demonstrates a meticulous approach to training a language model, incorporating various techniques and libraries to optimize both performance and efficiency.

Finally, the code culminates in a harmonious fusion of technology and user experience through the integration of a user-friendly interface using Gradio. This interface serves as the gateway through which users can effortlessly engage with the model, uploading images and inputting Tamil text prompts with ease. Through intuitive design and seamless interaction, the interface democratizes access to artifact-related information, empowering users to explore and learn about cultural heritage in their native language.

7. Conclusion:

In conclusion, the project exemplifies the synergy between advanced NLP techniques and cultural heritage preservation. By transcending linguistic barriers and fostering cross-cultural engagement, the Multimodal Language Model represents a significant stride towards democratizing access to museum artifacts. Through meticulous data scraping, question collection, translation, finetuning, and deployment, the project underscores the transformative potential of technology in bridging cultures and preserving humanity's rich tapestry of heritage.