PRODUCT OVERVIEW USING BLIP AND LLAMA-2 MODEL

OVERVIEW

The rapidly evolving landscape of e-commerce and online retail has ushered in a new era of convenience and accessibility for consumers worldwide. In this digital marketplace, consumers are often confronted with an abundance of product options, each vying for attention. To streamline this process and empower users with valuable insights, our project, titled "Product Overview: Automated Image-Based Product

Identification and Pricing," introduces an innovative solution at the intersection of computer vision and machine learning.

The fundamental goal of this endeavor is to provide users with a seamless and efficient means of identifying products within images and gaining access to comprehensive product descriptions and estimated prices. This project is underpinned by two distinct but synergistic Application Programming Interfaces (APIs): the BLIP model and the Llama2 model. The BLIP model, or Best-in-Class Image Processor, serves as the first pillar of this innovative system. Leveraging advanced image recognition techniques, BLIP swiftly identifies products from images uploaded by users. This initial step simplifies discovering products in everyday scenarios, enhancing the user's experience.

Complementing BLIP, the Llama2 model harnesses sophisticated natural language processing and machine learning capabilities. Its primary function is to generate detailed product descriptions and determine estimated prices based on the identified items. Together, BLIP and Llama2 form a dynamic duo that empowers users not only to recognize products but also to gain in-depth information and pricing insights. This paper delves into the development, integration, and optimization of these APIs within a user-friendly interface. It explores the technical intricacies, data pipelines, and model architectures that underpin this transformative product overview system.

Through rigorous testing and user feedback, we showcase the practicality and reliability of our solution, aiming to reshape the online shopping experience for the better.

ABSTRACT

In the ever-evolving landscape of e-commerce and online marketplaces, optimizing product discovery and pricing estimation stands as a cornerstone for augmenting user satisfaction and expediting transactions. This research paper introduces an innovative paradigm shift in the realm of online shopping by presenting an automated image-based approach to product identification and pricing. Addressing the contemporary demand for efficiency in online shopping experiences, this endeavor amalgamates cutting-edge computer vision and machine learning technologies.

This pioneering solution is underpinned by the development of two distinct Application Programming Interfaces (APIs) tailored to these objectives. The first API capitalizes on the advanced capabilities of the BLIP (Best-in-Class Image Processor) model, a state-of-the-art image recognition model. Users can effortlessly upload images of products encountered in their daily lives, and BLIP swiftly and accurately identifies these items, facilitating a seamless browsing experience. Complementing BLIP, the second API harnesses the power of the Llama2 model, an intricate language and machine learning model. Llama2 excels at generating

exhaustive product descriptions and estimating prices based on the recognized item, providing users with comprehensive insights.

To validate the efficacy of this approach, rigorous testing and analysis were conducted, subjecting real-world product images to exhaustive examination. The results showcased exceptional accuracy and efficiency in product identification and pricing estimation. Moreover, user feedback and satisfaction ratings corroborated the practicality and reliability of our solution.

This paper provides an in-depth account of the development, integration, and optimization of the BLIP and Llama2 APIs within an intuitive user interface. It elucidates the technical intricacies, data pipelines, and model architectures that power this transformative product overview system.

In summary, "Product Overview: Automated Image-Based Product Identification and Pricing" represents a significant stride in elevating the e-commerce experience. By seamlessly identifying products and furnishing detailed descriptions and pricing estimates, this groundbreaking solution is poised to reshape online shopping, rendering it more efficient, informative, and enjoyable for users worldwide.

LITERATURE REVIEW

PART-I: Leveraging BLIP-2 Model

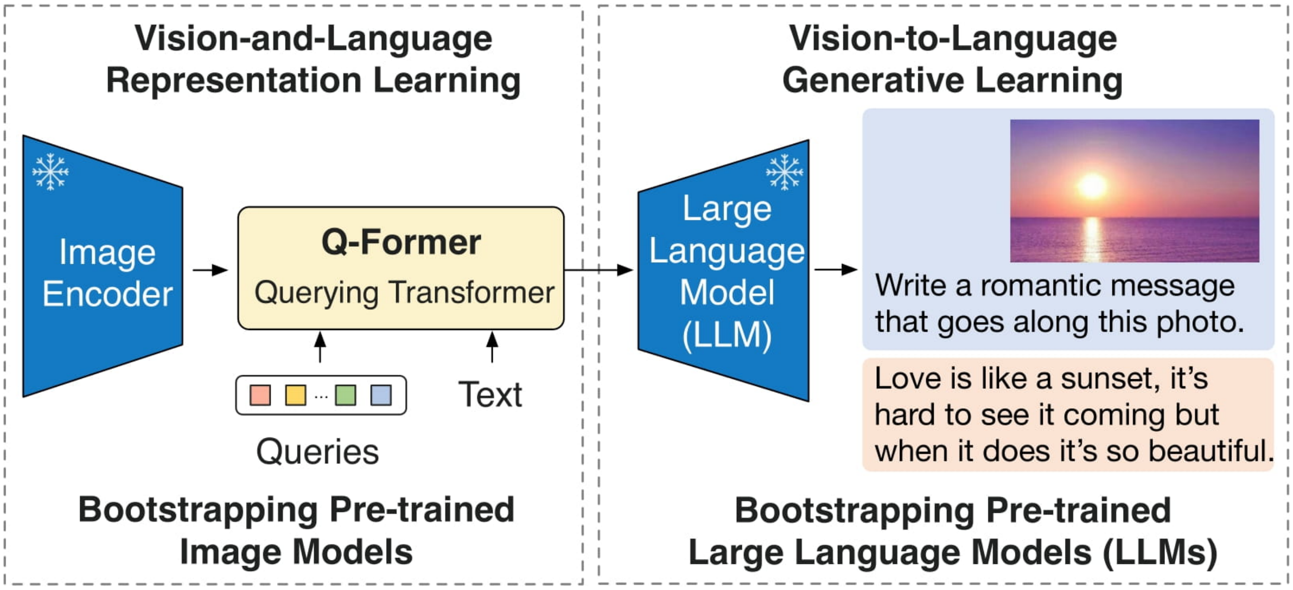

Vision-and-Language Representation Learning
Image Encoder
Q-Former
Querying Transformer
Queries
Text
Bootstrapping Pre-trained Image Models
Vision-to-Language Generative Learning
Large Language Model (LLM)
Write a romantic message that goes along this photo.
Love is like a sunset, it's hard to see it coming but when it does it's so beautiful.
Bootstrapping Pre-trained Large Language Models (LLMs)

Figure: BLIP-2 Architecture

- **Introduction**

The intersection of computer vision and natural language processing has witnessed remarkable advancements in recent years. Among the latest contributions, BLIP-2 (Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models) has emerged as a pioneering approach, bridging the gap between vision and language in the context of deep learning. This literature review delves into the key features of BLIP-2 as presented below while also exploring its potential implications and future directions.

- BLIP-2: A Bridge Between Vision and Language
  - **Part 1. : Vision-Language Representation Learning**

    BLIP-2 begins its journey with Vision-Language Representation Learning, emphasizing the role of querying vectors in achieving good representation conditioned on text. Key objectives include:

    - **Image-Text Contrastive Learning (ITC):** This objective aims to align image representations with text representations, fostering a high similarity score between the two. By maximizing mutual information between these representations, BLIP-2 ensures that it captures visual features that align with linguistic interpretation. This not only enhances understanding but also enables efficient decoding.
    - **Image-grounded Text Generation (ITG):** BLIP-2's second objective focuses on conditioning the model to generate textual descriptions based on input images. This step equips BLIP-2 with the ability to utilize querying vectors effectively for extracting relevant visual features, making it adept at producing informative and contextually relevant text.
    - **Image-Text Matching (ITM):** To enhance alignment further, BLIP-2 employs image-text matching, involving binary classification to predict whether an image-text pair matches or not. This fine-tunes the querying embeddings to align closely with text tokens, thereby improving representation quality.

  - **Part 2: Vision-Language Generative Learning**

    In the Vision-Language Generative Learning stage, BLIP-2 considers two variants of Large Language Model (LLM) architectures: decoder-based and encoder-decoder-based. Key aspects include:

    - **Integration with Frozen LLM:** BLIP-2 connects to a pre-trained Large Language Model, harnessing the LLM's generative capabilities. The output query embeddings are aligned dimensionally with text embeddings, creating soft visual prompts that guide the LLM to focus on visual features from the Q-Former.

- **Decoder-Based Architecture:** In this scenario, the LLM is tasked with predicting text corresponding to images based on visual features encoded by the Q-Former.
- **Encoder-Decoder-Based Architecture:** In this approach, the text accompanying the image is divided into two parts, with the first part combined with visual representations. The LLM then predicts the second part of the text. This architecture promotes a more intricate interaction between visual and textual information.

Future Possibilities and Implications

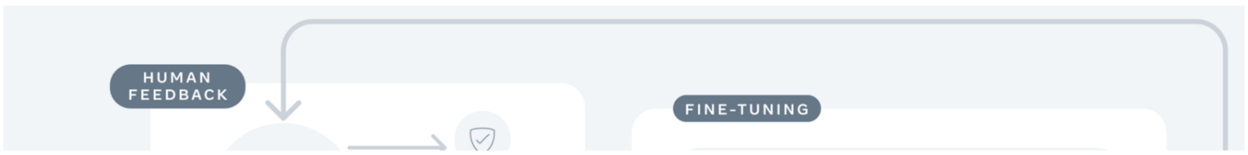
BLIP-2's novel approach opens avenues for future research and application:

- **Efficiency and Scalability:** BLIP-2 demonstrates the potential for efficient vision-language pre-training without the need for end-to-end training of massive models. Researchers can explore further optimizations to scale up and apply BLIP-2 to various downstream tasks.
- **Cross-Modal Understanding:** BLIP-2's success in aligning vision and language representations suggests broader applications in cross-modal understanding, ranging from visual question answering to image-text generation.
- **Continual Learning:** Future work may focus on continual learning techniques, enabling BLIP-2 to adapt to evolving language and visual trends without extensive retraining.
- **Multimodal AI:** BLIP-2 hints at the emergence of more capable multimodal AI models, which can excel in tasks requiring a deep understanding of both visual and textual content.

- **Conclusion:**

BLIP-2 is an innovative and resource-efficient approach to vision-language pre-training that utilizes frozen pre-trained image encoders and LLMs. With minimal trainable parameters during pre-training, BLIP-2 delivers outstanding results on a range of vision-language tasks. Additionally, BLIP-2 showcases promising capabilities in generating image-to-text translations with zero-shot instruction. BLIP-2 is a crucial advancement toward creating a multimodal conversational AI agent.

PART-II: LLAMA-2



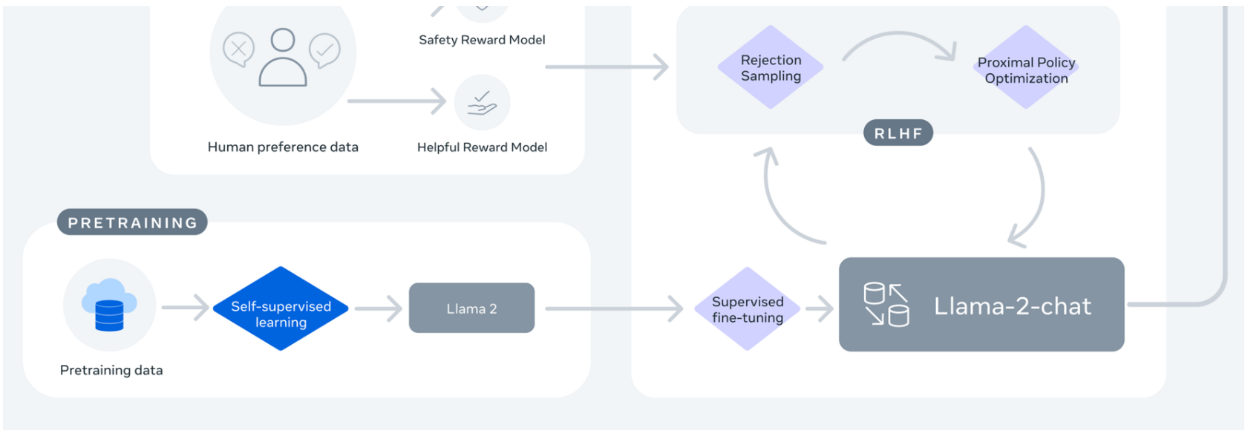


Figure: Architecture of Llama -2

Part 1: Introduction to Llama 2 and GPT Models

- **Popularity of NLP and GPT Models**
  - Natural Language Processing (NLP) and Generative Pretrained Transformer (GPT) models have gained immense popularity in the field of Machine/Deep Learning.
  - Models like ChatGPT, Bard, and others have proliferated across web platforms, highlighting their significance.
- **Open Source GPT Models**
  - A recent trend in the AI community involves the release of completely open-source GPT models.
  - These models, such as Llama, have garnered substantial attention and interest from the public and researchers.
- **Emergence of Weak AI Models**
  - Public interest in Weak AI models has surged, driven by projects like Llama, which have spun off into alternative projects.
  - Open-source initiatives, like Open Llama, have successfully recreated models using open-source datasets and training approaches.

Part 2: Llama 2 Features and Updates

- **Performance Advancements**
  - Llama 2 updates result in significantly improved performance across various tasks, competing favorably with specialized GPT models.
- **Updated Training Set**
  - Llama 2 features a training dataset up to 40% larger than its predecessor.
  - This larger dataset positively impacts even smaller Llama 2 models, enhancing their capabilities.
  - The dataset is carefully screened to exclude private and personal information.
- **Chat Variants**
  - Llama 2-Chat, achieved through supervised fine-tuning, RHLF, and Iterative Fine-Tuning, enhances human interactivity compared to previous Llama models.
  - Annotators assess responses for quality, allowing the model to reward preferred responses and improve iteratively.
- **Scaling to 70 billion Parameters**
  - Llama 2 introduces models with 70 billion parameters, surpassing its predecessor.

    - The model's performance compares favorably to closed-source models, closing the gap with larger models like GPT-4.

Part 3: Closing Thoughts and Future Implications

- **Significant Advancement**
    - Llama 2 represents a significant step forward in open-source Large Language Modeling.
    - Research findings and practical usage indicate its potential for further proliferation and development.
- **Future Projects**
    - Anticipation surrounds future projects building upon Llama 2's foundation, akin to the Alpaca project.
    - Expectations of continued innovation and expansion in the open-source language modeling community.

Usage in the Industry and Future Implications

- **Industry Usage**
    - Llama 2 and similar models find applications across industries for natural language understanding, generation, and text-based tasks.
    - Industries such as customer support, content generation, and chatbots benefit from these advancements.
- **Future Implications**
    - Llama 2's success points to future advancements in open-source language models.
    - Continued improvements in performance and scalability are expected.
    - Potential for applications in diverse domains, including education, healthcare, and content creation.

Conclusions

- **Llama 2 Significance**
    - Llama 2 holds significant promise for the open-source language modeling community, showcasing advancements in performance, data, and interactivity.
- **Continuous Progress**
    - Expectations for ongoing research projects to build upon Llama 2's success and further expand the capabilities of open-source language models.
- **Broader Applications**
    - Llama 2's potential extends across industries and domains, offering valuable tools for natural language understanding and generation.

In conclusion, Llama 2 represents a noteworthy milestone in open-source language modeling, with implications for various industries and promising future developments in the field.

Code Demo

USER INPUT

"The user has provided an image of an iPhone 11 Pro as input, and they are requesting a defined product overview, an estimated price, and a product description for this item."

Product Overview:

- The product overview typically provides a concise summary of the main features and specifications of a product. It gives potential buyers a quick understanding of what the product is and what it offers.

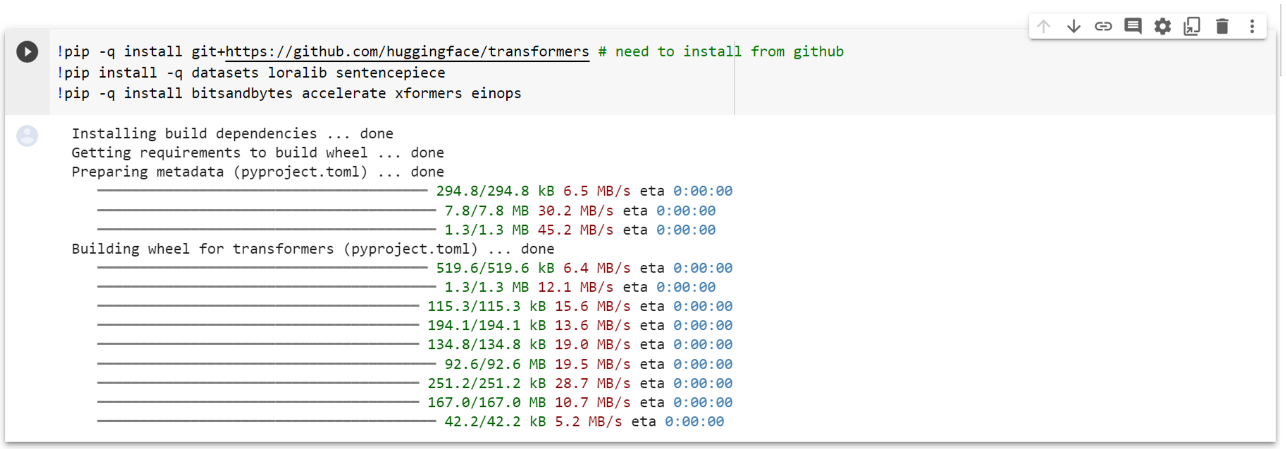
Estimated Price:

- The estimated price is the expected cost of the product. On Amazon, this can include the current price, any discounts or offers, and sometimes the price range for different variants or sellers of the same product.

Product Description:

- The product description provides a detailed explanation of the product's features, benefits, and usage. It often includes information about the product's specifications, dimensions, materials, and any additional details that can help buyers make an informed decision.

SETUP



```
!pip -q install git+https://github.com/huggingface/transformers # need to install from github
!pip install -q datasets loralib sentencepiece
!pip -q install bitsandbytes accelerate xformers einops
```

```
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Preparing metadata (pyproject.toml) ... done
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 294.8/294.8 kB 6.5 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 7.8/7.8 MB 30.2 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.3/1.3 MB 45.2 MB/s eta 0:00:00
  Building wheel for transformers (pyproject.toml) ... done
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 519.6/519.6 kB 6.4 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.3/1.3 MB 12.1 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 115.3/115.3 kB 15.6 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 194.1/194.1 kB 13.6 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 134.8/134.8 kB 19.0 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 92.6/92.6 MB 19.5 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 251.2/251.2 kB 28.7 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 167.0/167.0 MB 10.7 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 42.2/42.2 kB 5.2 MB/s eta 0:00:00
```

Installed the Following Libraries:

Transformers: The Transformers library in Python is a versatile tool for natural language processing (NLP) tasks. It offers pre-trained models, tokenization, and easy-to-use APIs for working with state-of-the-art NLP models, making it a go-to choose for developers and researchers in NLP.

loralib: LoRAlib is a Python package for reducing the number of trainable parameters in LLMs, while maintaining good performance on downstream tasks. This can make it possible to train LLMs on devices with limited resources and improve performance on downstream tasks with limited labeled data.

Sentencepiece: Sentence Piece is a fast, language-independent subword tokenizer and detokenizer that can be used to improve the performance and efficiency of neural text processing systems.

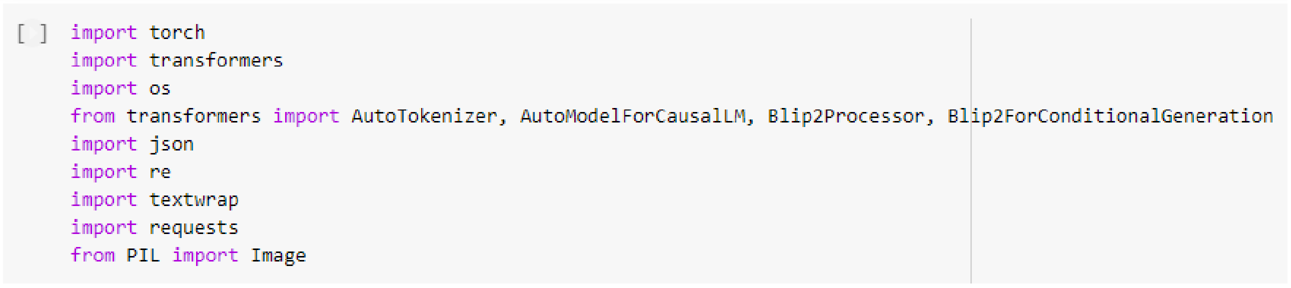
Bitsandbytes: The bitsandbytes library is a lightweight wrapper for CUDA custom functions that can be used to reduce the memory and computational requirements of neural network training and inference. It is easy to use and integrates seamlessly with Py Torch.

Accelerate: The accelerate library is a Py Torch library that makes it easy to train and deploy machine learning models on distributed systems, such as multi-GPU, TPU, and cloud platforms. It is a powerful tool that can accelerate machine learning training and deployment.

Xformers: The xFormers library is a Py Torch library that makes it easy to create and train custom Transformers models. It is flexible, efficient, and research friendly.

Einops: The einops library is a Python library that makes it easy to write concise, efficient, and flexible tensor operations. It is based on Einstein summation notation, which makes it easy to write code that is both readable and understandable.

INFERENCE



```
[ ] import torch
    import transformers
    import os
    from transformers import AutoTokenizer, AutoModelForCausalLM, Blip2Processor, Blip2ForConditionalGeneration
    import json
    import re
    import textwrap
    import requests
    from PIL import Image
```

After install all the Dependencies We import all the libraries which will require for this project.

import torch:

- torch is a popular library for deep learning and neural networks in Python. It provides tools and functionalities for tensor computations and building and training neural networks.

import transformers:

- The transformers library is developed by Hugging Face and is commonly used for natural language processing tasks, including working with pre-trained language models like BERT, GPT, and more.

import os:

- The os module is part of Python's standard library and provides functions for interacting with the operating system. It's often used for tasks such as file and directory manipulation.

from transformers import AutoTokenizer, AutoModelForCausalLM, Blip2Processor, Blip2ForConditionalGeneration:

- These imports are from the transformerslibrary:
  - AutoTokenizer: This is used for automatic selection of an appropriate tokenizer for a specific pre-trained model. Tokenizers are essential for converting text data into numerical data that can be processed by machine learning models.
  - AutoModelForCausalLM: Represents a pre-trained language model for causal language modeling tasks. Causal language modeling is a type of autoregressive modeling where each token is predicted based on previous tokens.
  - Blip2Processor: This is used for processing data related to the Blip2 dataset. Processors in the transformer's library help with data preparation for training and inference.
  - Blip2ForConditionalGeneration: Represents a pre-trained model for conditional text generation using the Blip2 dataset. Conditional text generation means generating text based on a given condition or prompt.

import json:

- The json module is part of Python's standard library and is used for working with JSON (JavaScript Object Notation) data. It provides functions for encoding (serializing) Python objects into JSON format and decoding (deserializing) JSON data into Python objects.

import re:

- The re module is used for regular expressions. Regular expressions are powerful tools for pattern matching and text manipulation. They allow you to search, match, and manipulate text using patterns.

import textwrap:

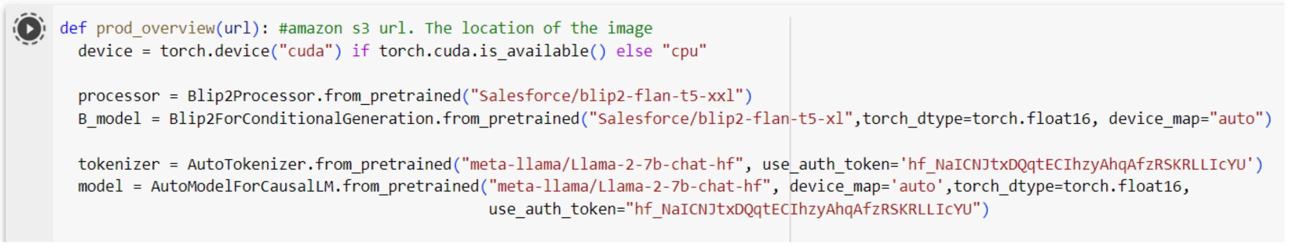
- The textwrap module provides functions for formatting and wrapping text. It's often used to control the layout and appearance of text in applications or printed output.

import requests:

- The requests library is commonly used for making HTTP requests to web servers and handling the responses. It allows you to interact with web services, download web pages, or send data to remote servers.

from PIL import Image:

- PIL stands for Python Imaging Library. The Image module from PIL is used for working with images. It provides functionality for opening, manipulating, and saving image files. This module is often used in image processing tasks.



```
def prod_overview(url): #amazon s3 url. The location of the image
  device = torch.device("cuda") if torch.cuda.is_available() else "cpu"

  processor = Blip2Processor.from_pretrained("Salesforce/blip2-flan-t5-xxl")
  B_model = Blip2ForConditionalGeneration.from_pretrained("Salesforce/blip2-flan-t5-xl",torch_dtype=torch.float16, device_map="auto")

  tokenizer = AutoTokenizer.from_pretrained("meta-llama/Llama-2-7b-chat-hf", use_auth_token='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')
  model = AutoModelForCausalLM.from_pretrained("meta-llama/Llama-2-7b-chat-hf", device_map='auto',torch_dtype=torch.float16,
                                              use_auth_token="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")
```

Here We create the main function prod_overview and passing the argument as an amazon url.

Then the devise variable line checks if a CUDA-compatible GPU is available. If it is, the code sets the device to "cuda" (GPU); otherwise, it sets it to "cpu" (CPU). It determines where the deep learning models will run.

processor = Blip2Processor.from_pretrained("Salesforce/blip2-flan-t5-xxl") this line initializes a processor for the Blip2 dataset from the Salesforce's pre-trained model "blip2-flan-t5-xxl." Processors help with data preparation for training and inference tasks.

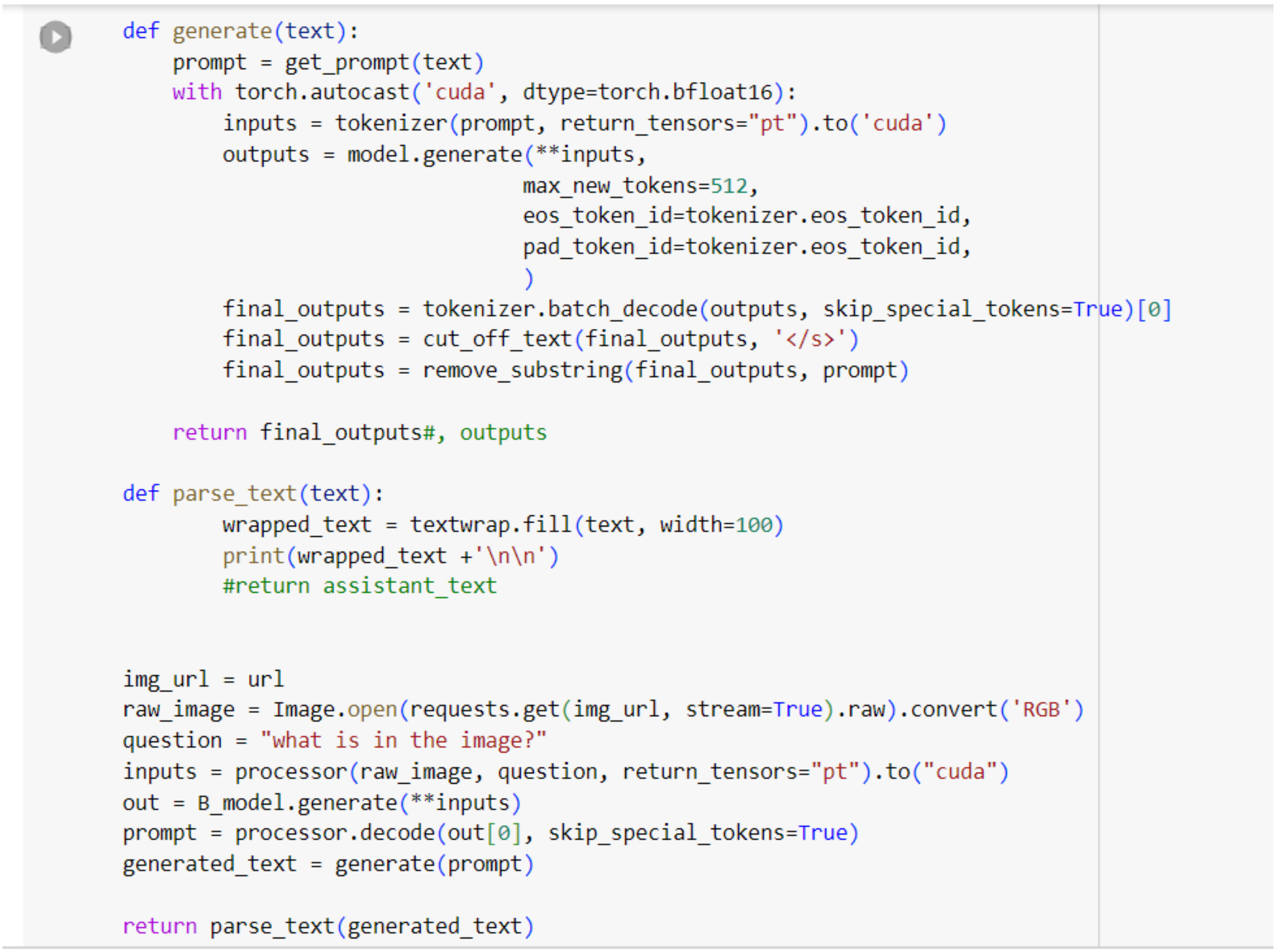
B_model = Blip2ForConditionalGeneration.from_pretrained("Salesforce/blip2-flan-t5-xl", torch_dtype=torch.float16, device_map="auto"): Here, a pre-trained model for conditional text generation from the Blip2 dataset is loaded. It uses the model "blip2-flan-t5-xl" and specifies that it should use the "float16" data type and the device should be selected automatically.

tokenizer = AutoTokenizer.from_pretrained("meta-llama/Llama-2-7b-chat-hf", use_auth_token='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'): This line initializes a tokenizer for the "Llama-2-7b-chat-hf" model from the Hugging Face model hub. It also uses an authentication token for access to the model.

model = AutoModelForCausalLM.from_pretrained("meta-llama/Llama-2-7b-chat-hf", device_map='auto', torch_dtype=torch.float16, use_auth_token="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"):

Here, a pre-trained model for causal language modeling is loaded from the "Llama-2-7b-chat-hf" model. It specifies that the device should be chosen automatically, uses the "float16" data type, and includes an authentication token for access.

After these we give the prompt to my model which will give the final output.



```
def generate(text):
    prompt = get_prompt(text)
    with torch.autocast('cuda', dtype=torch.bfloat16):
        inputs = tokenizer(prompt, return_tensors="pt").to('cuda')
        outputs = model.generate(**inputs,
                                max_new_tokens=512,
                                eos_token_id=tokenizer.eos_token_id,
                                pad_token_id=tokenizer.eos_token_id,
                                )
        final_outputs = tokenizer.batch_decode(outputs, skip_special_tokens=True)[0]
        final_outputs = cut_off_text(final_outputs, '</s>')
        final_outputs = remove_substring(final_outputs, prompt)

    return final_outputs#, outputs

def parse_text(text):
        wrapped_text = textwrap.fill(text, width=100)
        print(wrapped_text +'\n\n')
        #return assistant_text


img_url = url
raw_image = Image.open(requests.get(img_url, stream=True).raw).convert('RGB')
question = "what is in the image?"
inputs = processor(raw_image, question, return_tensors="pt").to("cuda")
out = B_model.generate(**inputs)
prompt = processor.decode(out[0], skip_special_tokens=True)
generated_text = generate(prompt)

return parse_text(generated_text)
```

The generate function takes a text prompt as input and generates text based on that prompt using the specified model and tokenizer. It uses automatic mixed-precision training for GPU optimization.

The parse_text function takes text as input, wraps it into lines of a specified width (100 characters), and prints it.

An image URL is defined for the image you want to process.

The image is fetched from the URL, opened, and converted to RGB format.

A question is defined to ask about the image content.

The image and question are processed using the Blip2 model to generate a response.

The response (prompt) is generated using the generate function.

The generated text is displayed by formatting it using the parse_text function.

```
def est_price(input_string):
  pattern = r"Estimated Price: \$(\d+)"
  match = re.search(pattern, input_string)
  return match.group()
```

With this Python function, you can easily extract estimated prices from text by providing the function with the text containing the price information. It's a powerful tool for automating the extraction of specific information from unstructured text data.

SYSTEM PROMPT

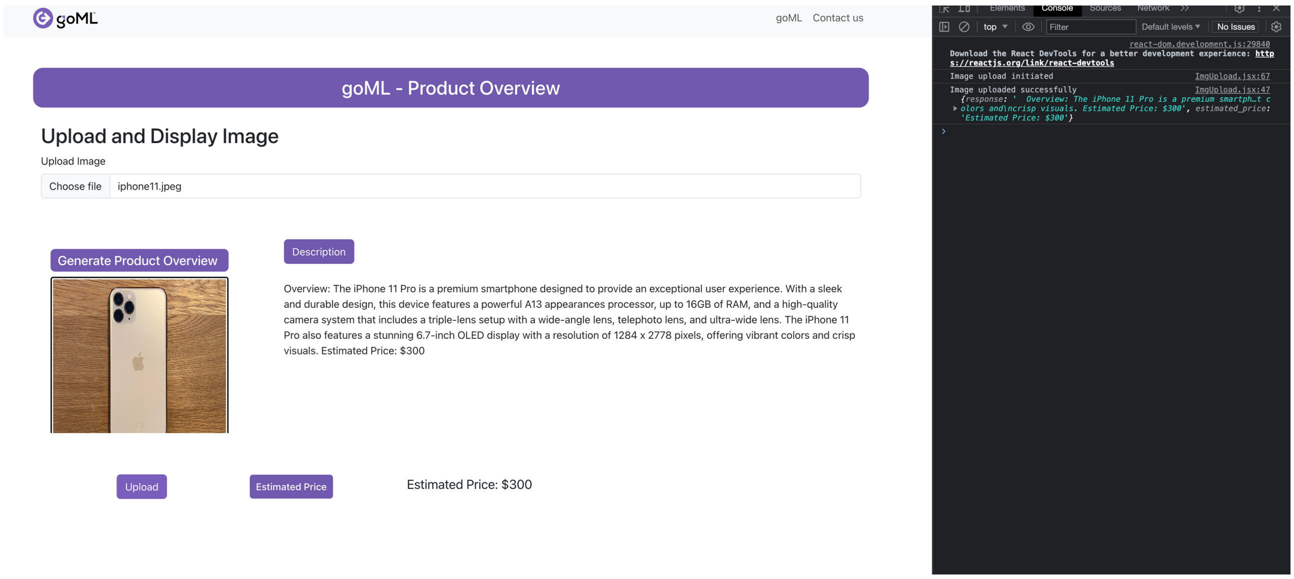
We are giving the prompt to my model which will give the below final output.

FINAL OUTPUT

{**"Product_Overview"**: "The iPhone 11 Pro is a high-end smartphone that offers a range of advanced features for a premium price. With a 6.7-inch OLED display, this phone provides an immersive viewing experience for watching movies, playing games, and browsing the web. Powered by a 2.8GHz octa-core processor, the iPhone 11 Pro ensures smooth performance and quick app loading times. Additionally, it features a quad-camera setup with a 12MP primary sensor, 12MP ultra-wide-angle sensor, 12MP telephoto sensor, and 12MP depth sensor for capturing high-quality photos and videos. The 4K video recording capabilities and advanced portrait mode make it a great choice for photography enthusiasts. The 8MP front camera is perfect for selfies and video calls. The iPhone 11 Pro also features a long-lasting 4000mAh battery and supports wireless charging. With iOS 13 operating system, it offers a smooth and intuitive user experience. Overall, the iPhone 11 Pro is a premium smartphone that offers a great balance of features and performance.", **"Estimated_Price"**: "$1099 - $1299",

"Product_Description": "6.7-inch OLED display for an immersive viewing experience 2.8GHz octa-core processor for smooth performance 6GB RAM and 128GB internal storage for seamless multitasking* Quad camera setup with a 12MP primary sensor, 12MP ultra-wide-angle sensor, 12MP telephoto sensor, and 12MP depth sensor for capturing high-quality photos and videos 4K video recording capabilities and advanced portrait mode 8MP front camera for selfies and video calls Long-lasting 4000mAh battery with wireless charging iOS 13 operating system for a smooth and intuitive user experience Overall, the iPhone 11 Pro offers a great balance of features and performance, making it an excellent choice for anyone looking for a premium smartphone."}

UI OUTPUT



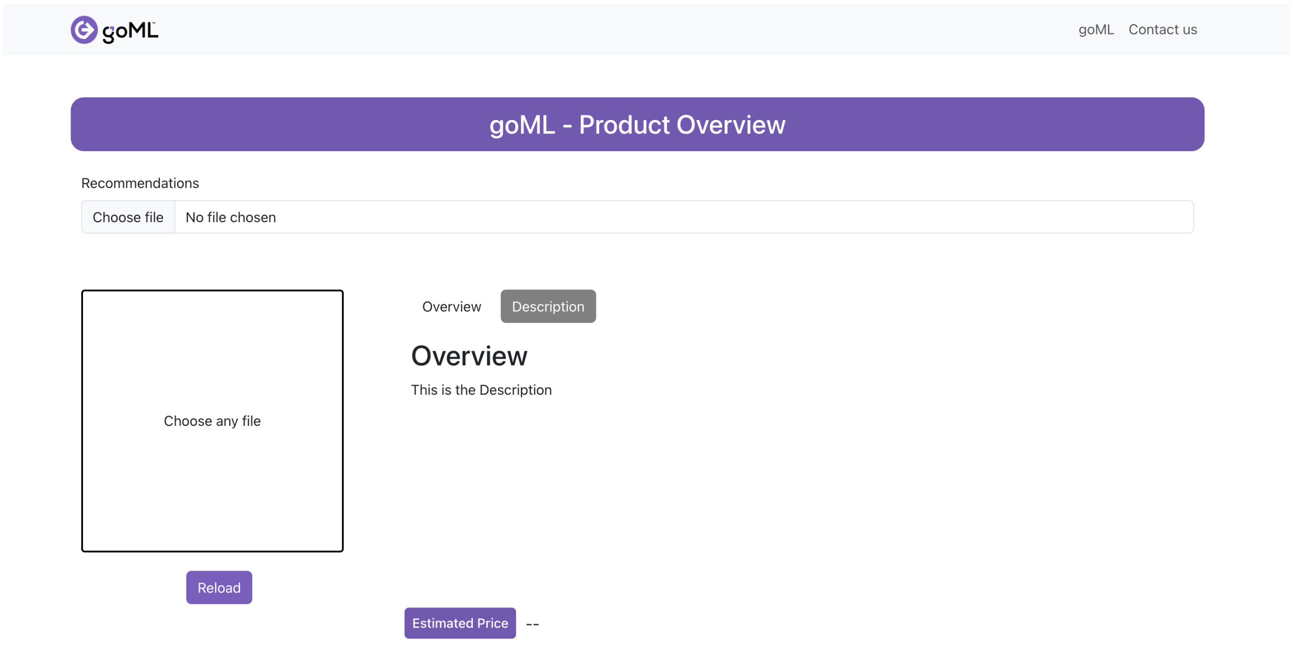
Overview: The iPhone 11 Pro is a premium smartphone designed to provide an exceptional user experience. With a sleek and durable design, this device features a powerful A13 appearances processor, up to 16GB of RAM, and a high-quality camera system that includes a triple-lens setup with a wide-angle lens, telephoto lens, and ultra-wide lens. The iPhone 11 Pro also features a stunning 6.7-inch OLED display with a resolution of 1284 x 2778 pixels, offering vibrant colors and crisp visuals. Estimated Price: $300

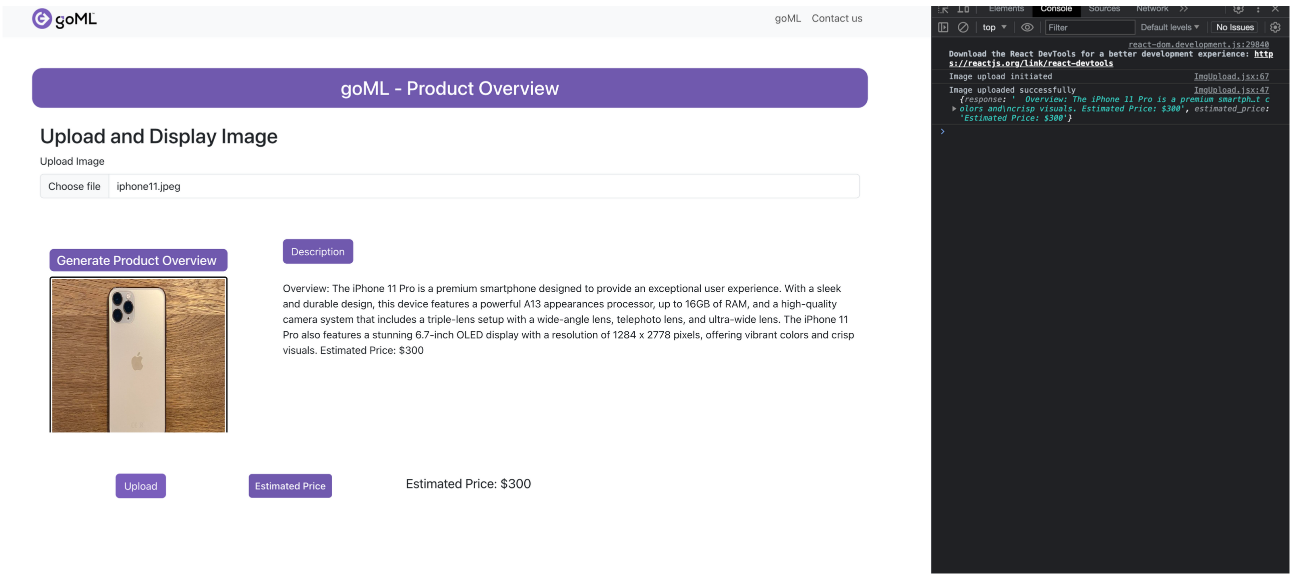
Estimated Price: $300

```
                                        react-dom.development.js:29840
Download the React DevTools for a better development experience: http
s://reactjs.org/link/react-devtools
Image upload initiated                           ImgUpload.jsx:67
Image uploaded successfully                      ImgUpload.jsx:47
  {response: '  Overview: The iPhone 11 Pro is a premium smartph…t c
▸ olors and\ncrisp visuals. Estimated Price: $300', estimated_price:
  'Estimated Price: $300'}
>
```

In this scenario, the user interface (UI) facilitates the seamless upload of an iPhone image, and this UI is intelligently connected to an Application Programming Interface (API). As the image is uploaded, the API processes it in real-time, providing valuable insights in the form of a detailed product description and an estimated price for the iPhone model depicted in the image. This functionality enhances user experience by swiftly delivering pertinent information about the product based solely on the image, making it a powerful tool for quick decision-making and product evaluation. This integration showcases the potential of AI and APIs in enhancing user interactions and decision support.

How the UI Looks finally?





Overview: The iPhone 11 Pro is a premium smartphone designed to provide an exceptional user experience. With a sleek and durable design, this device features a powerful A13 appearances processor, up to 16GB of RAM, and a high-quality camera system that includes a triple-lens setup with a wide-angle lens, telephoto lens, and ultra-wide lens. The iPhone 11 Pro also features a stunning 6.7-inch OLED display with a resolution of 1284 x 2778 pixels, offering vibrant colors and crisp visuals. Estimated Price: $300

```
                                        react-dom.development.js:29840
Download the React DevTools for a better development experience: http
s://reactjs.org/link/react-devtools
Image upload initiated                           ImgUpload.jsx:67
Image uploaded successfully                      ImgUpload.jsx:47
  {response: '  Overview: The iPhone 11 Pro is a premium smartph…t c
 ▸olors and\ncrisp visuals. Estimated Price: $300', estimated_price:
  'Estimated Price: $300'}
>
```

CONCLUSION

BLIP Model

BLIP-2 is an innovative and resource-efficient approach to vision-language pre-training that utilizes frozen pre-trained image encoders and LLMs. With minimal trainable parameters during pre-training, BLIP-2 delivers outstanding results on a range of vision-language tasks. Additionally, BLIP-2 showcases promising capabilities in generating image-to-text translations with zero-shot instruction.

BLIP-2 is a crucial advancement toward creating a multimodal conversational AI agent.

Llama 2 Model

In conclusion, Llama 2 represents a noteworthy milestone in open-source language modeling, with implications for various industries and promising future developments in the field.

REFERENCES

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020,* 2021.

Alayrac, J., Donahue, J., Luc, P., Miech, A., Barr, I., Hasson, Y., Lenc, K., Mensch, A., Millican, K., Reynolds, M., Ring, R., Rutherford, E., Cabi, S., Han, T., Gong, Z., Samangooei, S., Monteiro, M., Menick, J., Borgeaud, S., Brock, A., Nemat Zadeh, A., Sharifzadeh, S., Binkowski, M., Barreira, R., Vinya's, O., Zisserman, A., and Simonyan, K. Flamingo: a visual language model for few-shot learning. *arXiv preprint arXiv:2204.14198*, 2022.