SmartTool Guide: An AI Chatbot

Author:

Dharmagna Vyas

In the realm of Artificial Intelligence (AI), users often face challenges when attempting to navigate the extensive landscape of AI tools available. The lack of a centralized, user-friendly platform for discovering and learning about these tools hinders efficient exploration and utilization. To address this issue, the project aims to develop an AI Tools Discovery Chatbot, which serves as a conversational assistant for users seeking information about various AI tools.

```
# Step 1: Import necessary libraries
import pandas as pd
import numpy as np
import nltk
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
import tensorflow as tf
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity
import pickle

# Step 2: Load knowledge base data into a dictionary
knowledge_base_data = {
    'Category': ['Video Editing', 'Video Editing', 'Video Editing', 'Location', 'Research', 'Image Generation', 'Image Generation', 'Image Ge
    'Tool': ['Luma Labs', 'Skillshare', 'Chirp', 'Google Earth Studio', 'Siace', 'Google Search Generative Experience', 'Dream by WOMBO', 'Ar
    'Description': ['A tool that lets you turn a video on your phone into a fully interactive 3D scene.', 'A platform that offers a variety o
    'Link': ['https://lumalabs.ai/', 'https://www.skillshare.com/en/', 'https://www.youtube.com/watch?v=l-qg9ST1jJ8', 'https://www.google.com
}

# Step 3: Ensure all lists have the same length in the knowledge base data
min_length = min(map(len, knowledge_base_data.values()))
knowledge_base_data = {key: value[:min_length] for key, value in knowledge_base_data.items()}

# Step 4: Create a DataFrame using Pandas
df = pd.DataFrame(knowledge_base_data)

# Step 5: Download 'punkt' and 'stopwords' resources from NLTK
nltk.download('punkt')
nltk.download('stopwords')

# Step 6: Define a function for text preprocessing
"""
Tokenizes, stems, and removes stopwords from the input text.

Args:
    - text (str): Input text to be preprocessed.

Returns:
    - str: Preprocessed text.
"""
def preprocess(text):
    words = word_tokenize(text)
    words = [PorterStemmer().stem(word.lower()) for word in words if word.isalnum() and word.lower() not in set(stopwords.words('english'))]
    return ' '.join(words)

# Step 7: Define a function to preprocess knowledge base descriptions
def preprocess_descriptions(descriptions):

    return [preprocess(desc) for desc in descriptions]

# Step 8: Load knowledge base descriptions and preprocess them
knowledge_base_descriptions = preprocess_descriptions(df['Description'])

# Step 9: Create a TF-IDF vectorizer
```

```
tfidf_vectorizer = TfidfVectorizer(stop_words='english')
tfidf_matrix = tfidf_vectorizer.fit_transform(knowledge_base_descriptions)

# Step 10: Create an empty DataFrame to store chat responses
chat_responses_df = pd.DataFrame(columns=['User Input', 'Category', 'Tool', 'Description', 'Link'])

# Step 11: Define a function to preprocess user input and generate a response
def generate_response(user_input, df, tfidf_vectorizer, tfidf_matrix):

    user_input = preprocess(user_input)
    user_tfidf = tfidf_vectorizer.transform([user_input])

    similarities = cosine_similarity(user_tfidf, tfidf_matrix).flatten()
    most_similar_index = similarities.argmax()

    response = {
        'category': df['Category'][most_similar_index],
        'tool': df['Tool'][most_similar_index],
        'description': df['Description'][most_similar_index],
        'link': df['Link'][most_similar_index],
    }

    return response

# Step 12: Define the main chat loop
def chat():
    global chat_responses_df  # Declare it as a global variable

    print("Bot: Hi! I'm your AI chatbot. Ask me about AI tools, and I'll provide information based on the knowledge base.")

    while True:
        user_input = input("You: ")

        if user_input.lower() in ['exit', 'bye', 'quit']:
            print("Bot: Goodbye! Have a great day.")
            break

        response = generate_response(user_input, df, tfidf_vectorizer, tfidf_matrix)

        print(f"Bot: {response['category']} - {response['tool']}\nDescription: {response['description']}\nLink: {response['link']}")

        # Add the response to the DataFrame
        chat_responses_df = chat_responses_df.append({
            'User Input': user_input,
            'Category': response['category'],
            'Tool': response['tool'],
            'Description': response['description'],
            'Link': response['link']
        }, ignore_index=True)

# Step 13: Start the chat
if __name__ == "__main__":
    # Start the chat
    chat()

    # Save chat responses to Excel file
    chat_responses_df.to_excel('final_chat_responses.xlsx', index=False)
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
Bot: Hi! I'm your AI chatbot. Ask me about AI tools, and I'll provide information based on the knowledge base.
You: suggest me ai tools for productivity
<ipython-input-3-15572a920ec9>:100: FutureWarning: The frame.append method is deprecated and will be removed from pandas in a future ver
  chat_responses_df = chat_responses_df.append({
Bot: Productivity - Notion
Description: A product management tool that has an AI version that allows you to do things like set meeting agendas, write social media
Link: https://m.youtube.com/watch?v=29qGq2Ar5qY
You: suggest me image generation ai tool
<ipython-input-3-15572a920ec9>:100: FutureWarning: The frame.append method is deprecated and will be removed from pandas in a future ver
  chat_responses_df = chat_responses_df.append({
Bot: Image Generation - Google Search Generative Experience
Description: A tool that can generate images from text.
Link: https://www-google-com.cdn.ampproject.org/c/s/www.google.com
You: bye
Bot: Goodbye! Have a great day.
```

What I Learned:

Natural Language Processing (NLP):

Utilized NLTK for tokenization, stemming, and stopwords removal. Applied text preprocessing techniques to enhance the quality of input data. Information Retrieval:

Implemented TF-IDF vectorization to convert text data into numerical vectors. Calculated cosine similarity to find the most relevant tool based on user input. Chatbot Development:

Created a simple chatbot using Python and TensorFlow. Managed user interaction and provided meaningful responses based on the knowledge base. Data Handling with Pandas:

Utilized Pandas for data manipulation, including creating and updating DataFrames. Excel File Handling:

Saved chat history to an Excel file for future reference.

Project Summary

This project aims to develop an AI Tools Discovery Chatbot to simplify the exploration of a diverse range of Artificial Intelligence (AI) tools. The key challenges addressed include information overload, lack of personalized guidance, and the complexity of tool categorization. The chatbot utilizes Natural Language Processing (NLP) techniques for user queries, employing TF-IDF vectorization and cosine similarity to recommend relevant AI tools. Continuous updates to the knowledge base keep information current. User engagement is enhanced through detailed tool information and a history tracking feature. The scalable design accommodates future tool additions, and comprehensive documentation ensures maintainability. The AI Tools Chatbot promises to streamline the discovery and understanding of AI tools in a user-friendly and efficient manner.