Enhancing Language Learning with Multimodal AI & AR

AI-DRIVEN VIRTUAL PET FOR LANGUAGE LEARNING

Qi Sun, Yiyi Chen


Vocabulary acquisition
a lack of context, engagement, and personalization.
Traditional methods
may not address individual learner needs effectively.

OUR VISION: LEARNING BY LIVING

- In time
- interactive
- personalized
- immersive
- memorizing

Seamlessly integrate real-world objects and sounds into language learning.

Use personalized vocabulary levels to tailor content to the user.

Memory and test modes to reinforce learning effectively.

OUR VISION: LEARNING BY LIVING

- In time
- interactive
- personalized
- immersive
- memorizing

AI-Driven Virtual Pet for Language Learning

Acts as a friendly companion for real-time interactions.

Communicates with users through text, voice, and visual prompts.

Provides encouragement, tips, and reminders to enhance engagement.

Features and Modes

Visual Mode

UPLOAD/TAKE PHOTOS, ANALYZE CONTENT, AND GENERATE VOCABULARY.

INTERPRET IMAGE CONTENT, GENERATING DESCRIPTIONS, DEFINITIONS,

AND USAGE EXAMPLES ALIGNED WITH THE USER'S VOCABULARY LEVEL

Audio Mode

RECORD/UPLOAD SOUNDS AND GENERATE RELEVANT LEARNING CONTENT.

Random Chat

ASKING ANY QUESTION AND GET IMMEDIATE AND HELPFUL FEEDBACK FROM GPT.

Memory/Test Mode

REVIEW PAST INTERACTIONS AND MEASURE PROGRESS.

INTERACTIVE QUIZZES TO REINFORCE VOCABULARY RETENTION.

PROGRESS TRACKING TO MONITOR ADVANCEMENT AND ADAPT CONTENT.

AR and **AI** Integration ——Backend Technologies

AR

INTERACTIVE OVERLAYS ON REAL-WORLD OBJECTS AND SCENES.
VISUAL: YOLO, OPENCV, TENSORFLOW FOR OBJECT DETECTION.
AUDIO: PRE-TRAINED MODELS FOR SOUND CLASSIFICATION.

LLMs (OpenAI GPT APIs)

ANALYZE MULTIMODAL DATA TO GENERATE DESCRIPTIONS AND PERSONALIZED OUTPUTS.

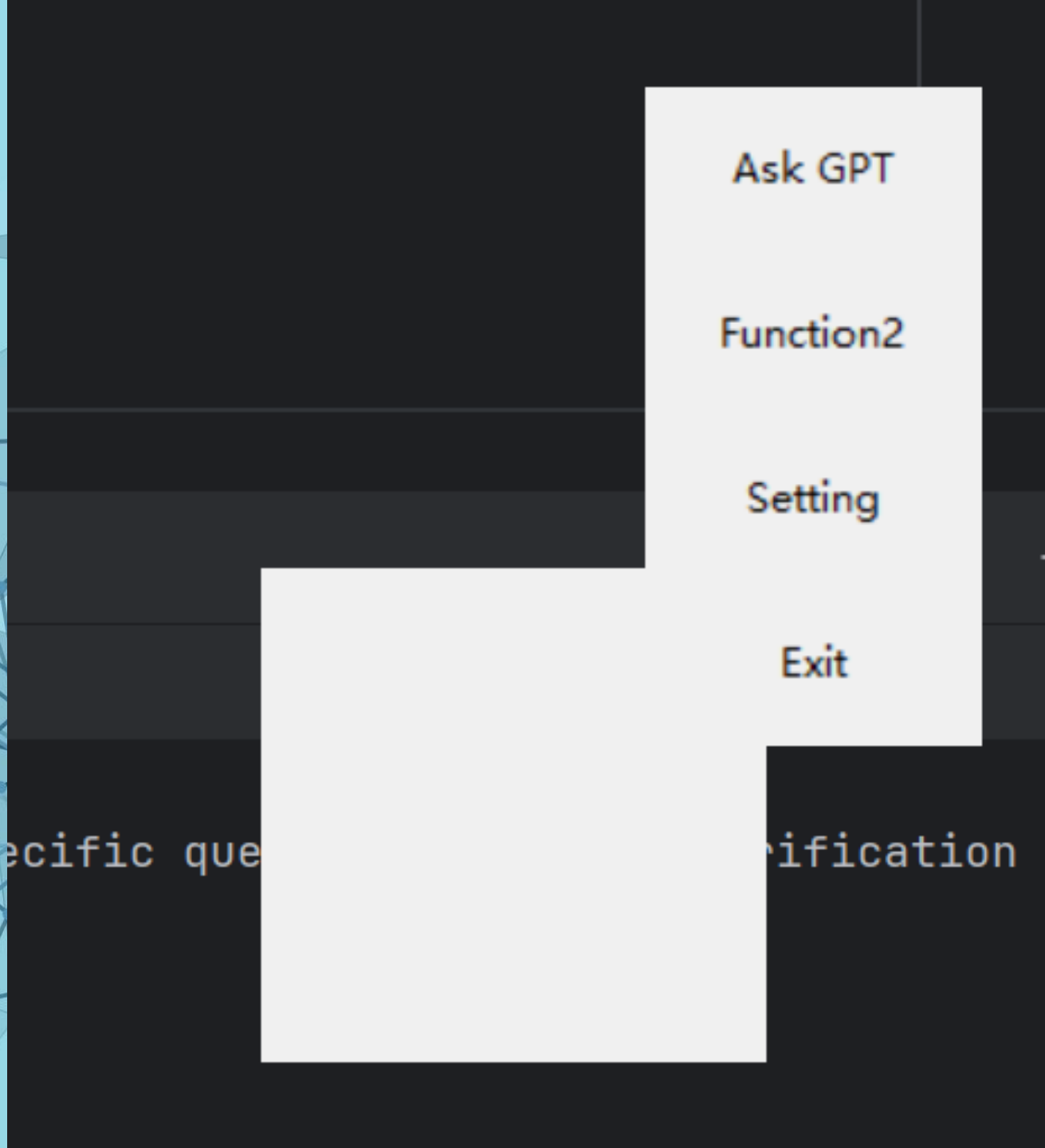
ADJUSTS VOCABULARY COMPLEXITY BASED ON USER PROFICIENCY

Virtual Pet Interaction

FINE-TUNED DIALOGUE GENERATION FOR INTERACTIVE AND ADAPTIVE COMMUNICATION.
EMOTIONAL INTELLIGENCE ALGORITHMS FOR MOTIVATING USERS.

DEMO

- real-time
- interactive
- customized
- multi-methods
- memorizing



3. **Constructor Details**:
   - `self.root`: Initializes the main window of the application by calling `main_stream()`.
   - `self.Image`: Set to `None`, likely meant to hold an image later.
   - `self.add_drag_functionality(self.root)`: This line suggests that there enable dragging interactions within the GUI.
   - `self.client`: Initializes the OpenAI client for interacting with the
   - `self.gpt_window_open`: A flag set to `False`, possibly to track if a GPT (presumably an AI model) is open.
   - `self.img`: Another variable presumably for handling images.
   - `self.default_prompt`: Sets a default prompt message for the applicati it's aimed at providing personalized learning assistance.

4. **Functionality**:
   - Although the complete code is not visible, the class appears to be gea an interactive application, possibly leveraging AI to assist users in learn tasks.

Overall, this structure indicates a framework for a graphical application t

Upload Image

Sample output for image chat

```python
import tkinter as tk
import json
from openai import OpenAI
from tkinter import filedialog
from PIL import Image, ImageTk
from tkinter import PhotoImage
import base64
from io import BytesIO


1 usage
class Pet:
    def __init__(self):
        self.root = self.main_stream()
        self.Image = None
        self.add_drag_functionality(self.root)
        self.client = OpenAI()
        self.gpt_window_open = False
        self.img = None
        self.default_prompt = 'You are a very helpful personalized learning helper to help student to learn better. Since

    1 usage
    def main_stream(self):
        root = tk.Tk()
        root.overrideredirect(True)
```