Meeting Recording #3

Retrieval Augmented Generation (RAG) and Model Communication

Introduction to APIs and Model Interacti on

- Discussion on accessing Large Language Models (LLMs) through APIs.
- Mention of specific tools/APIs:
  - Olama (O L L A M A)
  - llama.cpp (Wama CPP)
  - OpenRouter (alternative to self-hosted solutions)
- Emphasis on understanding how to structure API calls and enable communi cation between APIs and LLMs (e.g., Llama).
- Key areas for focus:
  - API structure
  - Data sharing via API
  - Setting up and building APIs

Retrieval Augmented Generation (RAG) - The Founding Paper

- **Paper:** "Retrieval Augmented Generation for Knowledge-Intensive NLP Tas ks" (May 2020)
- **Significance:** Introduced the concept of Retrieval Augmented Generation (RAG).
- **Core Problem Addressed:** Grounding LLMs to prevent hallucinations.
- **Historical Context:** Considered "ancient" in LLM terms (6 years old).

RAG Architecture (Original Method)

- **Components:**
  - **Encoder Model:** Processes documents to create vector representations. Encodes for semantics/meaning, not just words.
  - **Generator Model:** Produces the final output.
- **Process:**
  1. Run a similarity search (cosine or dot product) on vectors from the encod er.
  2. Rank documents by similarity.

3. Pass the actual document content to the generator.

- **Advantage:** Finds relevant information semantically, unlike traditional keyw ord-based searches (e.g., basic Google search, SQL queries).

Variants and Advancements in RAG

1. Graph RAG (Microsoft)

- **Mechanism:** Uses the model for preprocessing data to create a knowledge graph.
- **Process:**
  1. Model generates summaries of documents.
  2. Model identifies relationships between documents.
- **Output:** Provides summaries and associated documents, allowing traversal of the knowledge graph (nodes and edges).
- **Benefits:**
  - Significantly reduces query time.
  - Increases response accuracy.
- **Analogy:** Similar to the visualization feature in Obsidian, where notes (node s) are linked by relationships (edges).
- **Status:** Currently a highly peer-reviewed and adopted solution.

2. Recursive Language Models

- **Recency:** Emerged very recently.
- **Mechanism:** Does not involve model retraining. Instead, it finds information within already provided data.
- **Process:**
  1. Jams documents and prompt history into a Python REPL (Read-Eval-Print Loop) environment.
  2. This makes the data queryable using tools.
  3. The model receives the prompt, context, available tools, and previous doc uments.
  4. It iterates, asks questions, and performs follow-ups using the provided da ta and tools to answer the user query.
- **Advancement:** Handles massive context windows and iterates on informati on retrieval.
- **Benefit:** Significantly increases model accuracy compared to models with li mited context.
- **Example Context Size:** Can handle content equivalent to a novel.

Implementation Considerations for the Project

- **Challenge:** Object recognition for plant identification requires images, whic h is resource-intensive and prone to errors.
- **Proposed Solution:**
  - Allow the user to select the plant species.
  - Query a database for specific information about the selected plant.
  - Provide this contextual information to the LLM.
- **RAG Application:**
  - Directly grounding the model with database information.
  - Likely does not require a cosine similarity search as the relevant data is dir ectly retrieved.
- **Future Directions:**
  - Implementing image detection for plant identification (resource-intensiv e).
  - User selection is a more feasible approach given resource limitations.

Assessing Plant Health and Stress

Method 1: RGBD Camera and Deep Learning for Bi omass Prediction

- Utilizes an RGBD (color + depth) camera to capture images of plants.
- Employs deep learning to predict plant weight without harvesting.
- **Study Details:**
  - Experiment conducted on ~4000 lettuce plants.
  - Images captured at regular check sessions.
  - Plants were harvested and weighed immediately after imaging for data.
- **AI Capabilities:**
  - Predicts future plant weight.
  - Detects plant stress (e.g., nutrient deficiency).
- **Benefits:**
  - Calculates biomass without harming the plant.
  - Enables prediction of plant needs (nutrients, etc.).
- **Results:**
  - 7.3% inaccuracy in predictions.
  - Detected nutrient stress 8 hours to 2 days before visual symptoms appea red.
- **Potential Application:** Integrate deep learning model data if accessible to predict stress levels in project plants.
- **Limitation:** Primarily tested on lettuce.

Method 2: Chlorophyll Fluorescence

- Measures energy emitted by plants during photosynthesis.

- Different stress levels cause plants to emit varying levels of chlorophyll fluo rescence.
- **Traditional Method:** Uses a fluorometer costing $10,000+.
- **Cost-Effective Alternative:**
  - Used a monochromatic camera and a specific sheet (details not recalled).
  - Setup cost ~$200.
  - Achieved 0.03% inaccuracy compared to standard fluorometers.
- **Detection:**
  - Primarily detects diseases in plants.
  - Less affected by environmental factors like humidity.
- **Benefit:** Early stress detection, as symptoms appear later.

Method 3: Spatiotemporal Image Data and AI

- Similar to the RGBD method but uses 5 health scales.
- **Process:**
  - 12,000 images used.
  - A plant physiology expert scaled images weekly (levels 1-5).
  - This data (images + expert ratings) was fed to an AI.
- **AI Functionality:** Learns to predict plant health based on image data and e xpert labels.
- **Accuracy:** 82-85% accurate compared to expert assessments.
- **Application:** Users can take photos of plants for AI assessment of health.

Method 4: Plantix App (Disease Recognition & Rec ommendations)

- An available mobile application designed for farmers, particularly in Banglad esh.
- **Functionality:**
  - Recognizes plant diseases from leaf photos.
  - Provides recommendations for action.
- **Training Data:** Trained on 30,000+ images covering 35 diseases in 7 major crops.
- **Accuracy:**
  - Up to 97% on lab data.
  - 82-94% on real farm images.
- **Underlying Technology:** Utilizes GPT (likely large language model capabili ties) for image analysis and recommendations.
- **Potential Use:** Could be a prototype for smaller projects, though its databa se is limited and tailored to specific regions/crops.
- **Challenge for Small Projects:** Many agricultural solutions are designed for large-scale farming and may not easily adapt to smaller, indoor projects.

Narrowing Scope to Indoor Projects

- Initial research included broad agricultural applications, but a pivot to indoo r gardening was necessary for smaller project scope.
- Some methods blend crop-specific and indoor applications.

Method 5: Smart Flower Pot with 3D Printing

- A "smart flower pot" created using 3D printing.
- **Features:**
  - Automated watering.
  - Mobile app alerts for remote monitoring.
  - Built-in sensors: soil moisture, temperature, light.
- **Benefits:**
  - Alerts busy individuals to check plants remotely.
  - Supports the focus on moisture, temperature, and light as key indicators o f plant health over nutrients.
- **Status:** Peer-reviewed and considered promising.

Method 6: Indoor Monitoring using IoT

- Utilizes an Internet of Things (IoT) system, specifically a Raspberry Pi.
- **Functionality:**
  - Monitors indoor houseplants for moisture, temperature, and humidity.
  - Sends push notifications (e.g., via Pushbullet) to remind users about wate ring or repositioning plants.
- **Similarity:** Very similar in concept to the smart flower pot but uses a Raspb erry Pi system.

Method 6: Indoor Monitoring using IoT (Continued)

- **Hardware:** Uses a Raspberry Pi.
- **Sensors:** Soil moisture, light, and temperature/humidity sensors.
- **Functionality:**
  - Monitors plant conditions.
  - Sends push notifications to remind users to water or move plants.
- **Comparison:** Very similar to the smart flower pot project, also utilizing a Ra spberry Pi and sensors.

(End of lecture segment)