EXTRA CREDIT

GROUP SUBMISSION
ARCH DATA NETWORK INNOVATION SHOWCASE

REMEDYGENIE

AFFORDABLE AI FOR NATURAL HOME REMEDIES

Our project focuses on fine-tuning a Large Language Model (LLM) to provide safe, natural home remedies for common, non-serious illnesses. By using prompt engineering and the Unsloth library, we efficiently trained the model with data generated via Google Gemini. The goal is to create an affordable, AI-powered solution that delivers quick, reliable wellness advice, making healthcare support more accessible and aligned with the growing interest in natural treatments.

Prepared by: Harika Pamulapati
Nandini Yadav Gudala
Kavya Reddy Gondhi
Sangameshwar Ryakala
Murali Krishna Enugula

Professor: Adam Doyle
HDS-5230 High Performance Computing
Saint Louis University

Abstract
In the United States, healthcare costs are often prohibitively high, making access to routine or minor medical consultations difficult for many individuals. To address this challenge, our project aims to provide a low-cost, accessible, and AI-powered solution for everyday health concerns using natural home remedies. By fine-tuning a Large Language Model (LLM) on a carefully curated dataset of safe, household-based treatments for common, non-serious illnesses, we enable users to receive quick, natural remedy suggestions. Leveraging the Unsloth framework for efficient model training, LoRA for lightweight adaptation, and Gradio for an intuitive web interface, this one-stop solution empowers users with easy access to safe and practical health advice, minimizing cost without compromising usefulness.

Introduction
This project focuses on fine-tuning Large Language Models (LLMs) using the Unsloth framework to create a specialized AI assistant that provides safe, natural home remedies for common, non-serious illnesses. The primary goal is to train a model that can respond to user queries with cautious, household-based suggestions using ingredients like honey, ginger, turmeric, and more without recommending prescription medication.
To support this, the project automates the generation of a structured dataset using the Google Gemini API, producing diverse user prompts and carefully crafted remedy responses. The data is formatted in JSON Lines (.jsonl), with each entry containing:

- an instruction,
- a user input (symptom or illness), and
- an output (the remedy with disclaimers).

This dataset is then used to fine-tune a base model via Unsloth, resulting in an AI that can act as a health-aware, helpful assistant.

Definitions

1. **LLM (Large Language Model):**
   A **Large Language Model** is a type of AI that understands and generates human-like text. It's trained on massive amounts of text from the internet to learn patterns in language, so it can answer questions, write stories, chat, and more.
2. **Fine-Tuning:**
   Fine-tuning is like **teaching a smart AI a new specialty**. It takes a pre-trained LLM and trains it a bit more using your own custom data, so it gets better at specific tasks, like giving home remedies in your case.
3. **Unsloth:**
   Unsloth is a tool that makes fine-tuning large language models **faster and more memory-efficient**, especially on smaller GPUs. It’s designed to help people train big models without needing expensive hardware.
4. **LoRA (Low-Rank Adaptation):**
   LoRA is a smart technique used during fine-tuning that **reduces the amount of**

training needed. Instead of changing the whole model, it only adjusts small, specific parts, saving time, memory, and compute resources.

5. **SFTTrainer:**
   A tool from Hugging Face that helps fine-tune language models efficiently by training them on example input-output pairs, focusing on smaller parts like LoRA adapters to reduce memory and speed up the process.
6. **Gradio:**
   Gradio is a Python library that lets you build **simple web apps for AI models**. You can create a user interface (UI) to let people interact with your model through a browser, like typing an illness and getting a home remedy.
7. **Hugging Face:**
   Hugging Face is a platform and community for working with machine learning models, especially language models. It provides tools to share models, datasets, and code, kind of like GitHub, but for AI.

Implementation

1. **Setup & Unsloth Installation (Cells 1-3):** This step installs the Unsloth library and its latest updates from GitHub. Unsloth provides optimizations to make loading and fine-tuning large language models significantly faster and more memory-efficient, preparing the Colab environment.
2. **Base Model Loading (Cell 4):** Loads the specified base language model (unsloth/Llama-3.2-3B-Instruct) using Unsloth's FastLanguageModel. It applies 4-bit quantization (load_in_4bit = True) to drastically reduce memory requirements, making it feasible to run on a free Colab GPU.
3. **LoRA Adapter Setup (Cells 5-6):** Applies Low-Rank Adaptation (LoRA) to the loaded base model. This inserts small, trainable "adapter" layers into the model, allowing efficient fine-tuning by updating only these adapters instead of the entire model.
4. **Data Generation (Cells 7-14):** Uses the Google Gemini API to automatically generate the training data. It prompts Gemini to create a list of common illnesses and then pairs each illness with multiple generated home remedy suggestions (ensuring uniqueness), saving the results into the home_remedies_dataset.jsonl file.
5. **Data Preparation (Cells 15-16):** Loads the generated home_remedies_dataset.jsonl file. It then formats each example (instruction, input, output) into the specific "Alpaca" prompt template required by the instruct-tuned model and the training library, ensuring necessary special tokens (like the End-of-Sequence token) are added.
6. **Model Training (Cells 17-21):** Configures and executes the fine-tuning process using Hugging Face TRL's SFTTrainer. It trains the LoRA adapters (only ~0.8% of total parameters) on the prepared home remedies dataset for a small number of steps (30) using memory-efficient settings like 8-bit AdamW and mixed-precision.
7. **Inference Demonstration (Cells 22-25):** Shows how to use the fine-tuned model for generating text. It specifically demonstrates prompting the model with an illness (e.g., "Sore throat") using the correct format and uses a TextStreamer to display the generated home remedy suggestion token by token as it's created.

8. **Model Saving (Cells 26-35):** Saves the results of the fine-tuning. First, it pushes the lightweight LoRA adapters to Hugging Face Hub. Then, it merges these adapters with the base model, converts the merged model to the GGUF format (specifically Q8_0 quantization), and uploads this GGUF version to the Hub for use in llama.cpp-compatible applications.
9. **Gradio Web Interface (Cells 38-41)**: Installs the Gradio library and creates a simple, interactive web application. This UI loads the fine-tuned model (from the GGUF Hub repository), allows a user to type in an illness, and then streams the generated home remedy suggestion back to the user in real-time within the web interface.

Methodology

We began by installing the necessary libraries, including Unsloth, which helps load and fine-tune large language models quickly and efficiently, even on free Google Colab GPUs.
Next, we selected Llama-3.2-3B as our base model and applied LoRA (Low-Rank Adaptation). LoRA is a technique that adds small trainable layers to the model. Instead of updating the entire huge model, we only train these small parts, saving memory and time while still customizing the model for our task.
We then used the Google Gemini API to automatically create a dataset of home remedies for common illnesses. Before training, this data was formatted properly (following the Alpaca-style format) so the model could learn from it correctly. The format depends on which LLM you're using.
The model was then fine-tuned using Hugging Face's SFTTrainer. During this, only the LoRA adapters were trained (not the full model), which makes the process fast and resource-friendly. These adapters help the model learn how to give good remedy suggestions.
Once trained, we used inference to test the model, this means prompting it with an illness and watching it generate a helpful home remedy response, step-by-step.
Finally, the model (including LoRA adapters) was saved and uploaded to Hugging Face Hub. We also merged the model and converted it to GGUF format for easier use in lightweight environments.
To make the model easy to use, we built a simple Gradio web interface. This lets users type in an illness and instantly get a natural remedy suggestion in their browser.

Conclusion

In conclusion, this project demonstrates how to efficiently fine-tune a large language model using Unsloth and LoRA to create a lightweight, cost-effective assistant that provides safe and natural home remedies. By combining automated data generation with Gemini, memory-efficient training using SFTTrainer, and a simple Gradio interface, we built an accessible solution that addresses the need for affordable health advice, especially in contexts like the U.S. where healthcare costs can be high.