Personalized Food Recipe Recommender System

Murun Enkhtaivan

Department of Computer Science, San Jose State University, San Jose, CA, USA

This paper introduces a novel approach to recommending personalized food recipes by fine-tuning Large Language Model and implementing Retrieval-Augmented Generation (RAG) to address diverse dietary needs.

***Index Terms*—Personalization, Food Recipes, Recommender System, Machine Learning, Fine Tuning, RAG Implementation, Natural Language Processing.**

I. Introduction

In the rapidly evolving field of culinary technology, personalized dietary recommendations are increasingly important, catering to the unique preferences and time constraints of individuals.

This project develops a personalized food recipe recommender system by fine-tuning the GPT-2 model from Hugging Face's Transformers library. The goal was to adapt this pre-trained model to better understand and generate culinary content, specifically recipes that could be tailored to individual preferences based on input ingredients. This fine-tuning process was critical for the model to learn the structure and style of recipe writing, focusing on the seamless integration of raw ingredients into coherent cooking instructions.

To further enhance the performance and relevance of the generated recipes, we incorporated a Retrieval-Augmented Generation (RAG) approach. This method combines the generative capabilities of GPT-2 with the retrieval of recipe information during the generation process, allowing the system to produce more accurate and contextually appropriate results.

The integration of RAG aimed to leverage the vast repository of culinary data more effectively, ensuring that the recommendations are not only quick to prepare, as indicated by the dataset's tags, but also varied and suited to diverse dietary needs.

A. Motivation

The burgeoning interest in applying artificial intelligence to everyday life extends significantly into the culinary world, where the demand for personalized dietary recommendations is rapidly increasing. The primary motivation behind this project was to explore the capabilities of fine-tuning large language models, specifically the GPT-2, on a custom dataset tailored for food recipes. This endeavor aimed to harness the model's advanced text generation capabilities to produce contextually relevant and personalized culinary recommendations based on user input.

Additionally, the project sought to investigate the implementation of Retrieval-Augmented Generation (RAG) techniques to enhance the outcomes of the fine-tuned model. By integrating RAG, the system not only generates text based on learned patterns but also dynamically retrieves relevant information during the generation process, thereby improving the accuracy and relevance of the recommendations. This hybrid approach underscores a significant exploration into improving AI-driven systems in real-world applications, demonstrating a practical use case of deploying sophisticated machine learning techniques to meet specific user needs in the domain of personal nutrition and culinary arts.

II. System Architecture and Implementation

This section outlines the architecture of the personalized food recipe recommender system, emphasizing the data handling, model training, and algorithmic approaches employed to enhance the system's recommendation capabilities.

A. Data Collection and Preprocessing

- **Dataset Description**:
  The project utilized a comprehensive dataset comprised of approximately 450,000 food recipes, each annotated with various tags that describe the dish, including preparation time, ingredients, and cooking steps. A specific focus was maintained on recipes tagged '30-minutes-or-less' to cater to users seeking quick meal solutions.
- **Data Collection**:
  The recipes were sourced from an extensive culinary database, ensuring a wide range of dishes to enhance the diversity of the training data. This variety is crucial for the system's ability to generalize across different culinary tastes and preferences.
- **Preprocessing Steps**:
  - **Sampling**: The dataset was randomly sampled to select a subset of 160,000 recipes, which helps in managing computational resources more efficiently while ensuring a representative mix of the data.
  - **Cleaning**: Data cleaning involved removing any incomplete entries and normalizing ingredient names to reduce variability in the dataset.
  - **Tokenization**: Ingredients and cooking steps were tokenized using a customized tokenizer adapted for culinary terminology, which is crucial for effective model training.

B. Algorithm Implementation

- **Fine-Tuning GPT-2**: The core of the recipe generation is based on fine-tuning the GPT-2 model, a pre-trained large language model known for its effectiveness in generating human-like text. The model was fine-tuned on the processed recipe dataset to adapt its capabilities to generate culinary content that is coherent and contextually relevant to the input provided by the user.
- **Retrieval-Augmented Generation (RAG)**:
  - **Retrieval Component**: Before generating text, the system retrieves the most relevant existing recipes based on the input ingredients using a vector similarity search. This retrieval is powered by embedding the recipes into a high-dimensional space where semantic similarities can be quantitatively assessed.
  - **Generation Component**: The retrieved recipes inform the generation process, allowing the model to produce recommendations that are not only inspired by the training data but also closely aligned with the retrieved content.
- **Hybrid Techniques**: The implementation did not rely on traditional collaborative filtering or content-based filtering. Instead, it combined the generative prowess of GPT-2 with the retrieval efficiency of RAG, forming a hybrid model that leverages both generative and retrieval-based techniques to produce personalized and contextually relevant recipes.

C. System Architecture Diagram

Below is the initial plan of action represented as a system architecture diagram, outlining the major components and data flow within the recommender system.

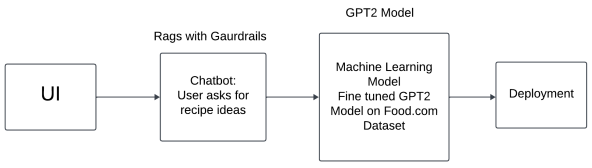

Figure 1: Initial System Architecture Plan

III. Fine-Tuning a Large Language Model on a Custom Dataset

Fine-tuning a pre-trained large language model like GPT-2 on a specialized dataset allows for leveraging the model's inherent linguistic capabilities while adapting it to specific contextual nuances of the target application—in this case, culinary recipes. This process enhances the model's ability to generate relevant and personalized content based on user input, making it a crucial step in developing an effective recipe recommender system.

A. Dataset Preparation and Preprocessing

Before fine-tuning the GPT-2 model, the dataset underwent crucial preprocessing to optimize model training conditions. This included random sampling, where a subset of 160,000 recipes was selected from the initial 450,000 to efficiently manage computational resources and ensure data diversity. Additionally, the data was cleaned to remove incomplete entries and standardize ingredient names, thereby enhancing model accuracy. Ingredients and cooking steps were also tokenized using a customized tokenizer adept at understanding culinary terminology, crucial for maintaining the contextual relevance of the generated recipes.

B. Model Configuration and Training

The GPT-2 model was configured with custom tokens to accommodate specific syntactic features of recipe text. The training process was set up as follows, using Hugging Face's `Transformers` and `Accelerate` libraries for efficient execution:

```
from transformers import GPT2TokenizerFast,
GPT2LMHeadModel
from transformers import Trainer, TrainingArguments

tokenizer = GPT2TokenizerFast.from_pretrained('gpt2',
bos_token='', eos_token='', unk_token='',
pad_token='')
model = GPT2LMHeadModel.from_pretrained('gpt2')
model.resize_token_embeddings(len(tokenizer))

# Define training arguments
args = TrainingArguments(

 output_dir='/content/drive/MyDrive/CS271_Project/Food
GPT',
    num_train_epochs=1,
    per_device_train_batch_size=2,
    save_strategy='no'
)

# Initialize the Trainer
trainer = Trainer(
    model=model,
    args=args,
    train_dataset=train_dataset,
    eval_dataset=eval_datase
)

trainer.train()
```

Figure 2: Code Implementation of Fine Tuning GPT-2 Model

C. Challenges and Solutions

One of the primary challenges in fine-tuning GPT-2 was adapting it to the specific style and structure of recipes, which are significantly different from the text types the model was originally trained on. To address this, extensive tokenization and embedding adjustments were made. Additionally, training on a niche dataset required careful handling of overfitting, managed through selective data sampling and rigorous evaluation during the training process.

This fine-tuning stage is critical as it tailors the pre-trained model to perform effectively within the specialized domain of culinary recipes, thereby significantly enhancing the quality and relevance of the generated recommendations in the recipe recommender system.

IV. Implementation of Retrieval-Augmented Generation (RAG)

The introduction of Retrieval-Augmented Generation (RAG) into the personalized recipe recommender system significantly enhances the model's ability to generate relevant and accurate recipe suggestions. RAG combines the power of a large language model (LLM) with a retrieval system to enrich the generation process with contextual data, thus producing more informed and precise outputs.

A. RAG Components

Knowledge Base Preparation: The RAG system begins with the preparation of a knowledge base, consisting of a comprehensive dataset of recipes. This dataset acts as a repository from which relevant information can be retrieved during the generation process.

Embedding and Indexing: Using a sentence transformer, each recipe in the knowledge base is transformed into a dense vector representation, creating embeddings that capture the semantic nuances of the culinary data. These embeddings are then indexed using FAISS (Facebook AI Similarity Search), which facilitates efficient retrieval of the most relevant recipes based on the query.

Retrieval Process: When a query is received, such as a list of ingredients or a specific dietary request, the system generates an embedding for the query and uses the FAISS index to retrieve the k-nearest recipes from the knowledge base. This retrieval step ensures that the generative model has access to contextually relevant data before producing the final recipe recommendation. The system retrieves the top five most similar recipes (k=5), ensuring that the generative model leverages contextually relevant information to produce accurate recipe recommendations.

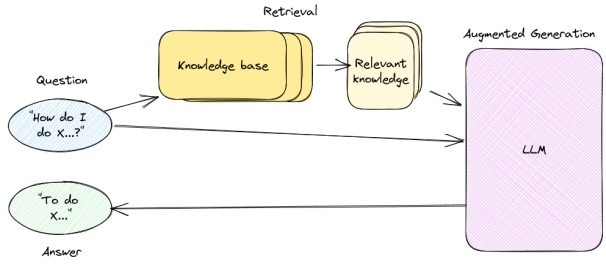

Figure 3: Overview of RAG

B. Integration with GPT-2

Generating Recipes: The retrieved recipes serve as augmented input for the fine-tuned GPT-2 model, which then generates a complete recipe. This process leverages the generative capabilities of GPT-2, enhanced by the contextual relevance provided by the RAG system.

C. Handling Data Type Errors with RAG Guardrails

During the implementation, we encountered data type errors, particularly when interfacing the RAG system with guardrails designed to validate the inputs and outputs. These errors were primarily due to discrepancies in expected data formats between the retrieval outputs and the generative model's inputs.

Solutions Implemented: To resolve these errors, we introduced type-checking mechanisms and format validators within the guardrails framework. This ensured that all data passed between the systems adhered to the required specifications, maintaining the integrity and functionality of the RAG pipeline. Adjustments were also made to the guardrails' configuration to explicitly recognize and handle the specific data structures used by our retrieval and generation components.

This comprehensive approach not only rectified the data type issues but also enhanced the system's robustness, ensuring that the recipe recommendations are both accurate and relevant.

V. Conclusion and Future Work

A. What We Learned

Throughout the course of this project, significant insights were gained into the capabilities and applications of Large Language Models (LLMs) such as GPT-2. The project provided a deep understanding of LLM architecture and demonstrated how they can be leveraged to generate text that is both coherent and contextually appropriate. We developed proficiency in fine-tuning these models on custom datasets, a crucial skill for adapting generic models to specialized tasks or industries. Moreover, the exploration into Retrieval-Augmented Generation (RAG), particularly the document-based RAG, expanded our knowledge on how a knowledge base can significantly augment the generative capabilities of LLMs. We also explored different methods for managing large datasets, including when to apply dimension reduction for feature simplification versus random sampling to reduce computational load while maintaining a representative dataset.

B. Future Directions

Building on the foundation laid by this project, several avenues are open for future exploration and enhancement. One immediate improvement would be to increase the number of training epochs, which could potentially improve the model's understanding of the dataset, leading to better quality and more nuanced recipe recommendations. Incorporating guardrails to ensure that the model's outputs meet certain ethical standards and guidelines is crucial, and learning how to effectively implement these guardrails will be a key focus. Another key focus will be placed on refining the implementation of Guardrails, specifically targeting the resolution of data format discrepancies. This effort will improve the accuracy and relevance of the outputs. Experimenting with different models, such as GPT-3 or other emerging models, could offer improvements in accuracy and generation capabilities. Finally, deploying the model using Google Cloud's Vertex AI presents a promising opportunity for scalability and enhanced user accessibility. This will not only help in managing the infrastructure more efficiently but also in scaling the application to handle more users.

In summary, this project not only enhanced our understanding of cutting-edge AI technologies but also set the stage for future innovations in the culinary technology space. The roadmap ahead is filled with opportunities to refine our system, experiment with new technologies, and ultimately, provide more value to users seeking personalized culinary experiences.

VI. References

1) C. N. Sekharan and P. S. Vuppala, "Fine-Tuned Large Language Models for Improved Clickbait Title Detection," 2023 Congress in Computer Science, Computer Engineering, & Applied Computing (CSCE), Las Vegas, NV, USA, 2023, pp. 215-220, doi: 10.1109/CSCE60160.2023.00039. keywords: {Deep learning;Training data;Transformers;Natural language processing;Data models;Task analysis;Machine learning;OpenAI API;Click-Bait Detection;LLM;Fine-tuning;Ada;GPT-4},
2) M. Goel et al., "Ratatouille: A tool for Novel Recipe Generation," 2022 IEEE 38th International Conference on Data Engineering Workshops (ICDEW), Kuala Lumpur, Malaysia, 2022, pp. 107-110, doi: 10.1109/ICDEW55742.2022.00022. keywords: {Deep learning;Conferences;Data engineering;Natural language processing;Data models;Novel Recipe Generation;Deep Learning;LSTM;Machine Learning;GPT2},
3) Bodhisattwa Prasad Majumder, Shuyang Li, Jianmo Ni, and Julian McAuley. Generating personalized recipes from historical user preferences. arXiv preprint arXiv:1909.00105, 2019.
4) K. Yanai, D. Horita and J. Cho, "Food Image Generation and Translation and Its Application to Augmented Reality," 2020 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR), Shenzhen, China, 2020, pp. 181-186, doi: 10.1109/MIPR49039.2020.00045. keywords: {Gallium nitride;Image generation;Generative adversarial networks;Training;Twitter;Image segmentation;Feature extraction;Food Image Generation;Food Image Translation},