Recipe Recommendation System Report

GROUP 15
SHIQI TU, ZIHAO LI, JIE LIN
LIANGYU CHANG, YUMO JIANG

Introduction

In today's fast-paced society, individuals often encounter difficulties in deciding what to cook, particularly when faced with constraints such as limited ingredients, time, or specific dietary preferences. This challenge frequently results in frustration and inefficient time spent searching for suitable recipes. To address this issue, our team proposes the development of a content-based recipe recommendation system designed to assist users in identifying recipes that align with the ingredients they have on hand, the time they are willing to spend cooking, and their individual taste preferences. This system aims to reduce the effort involved in recipe selection while maximizing ingredient utilization.

The primary objective of this system is to enhance the overall cooking experience by offering personalized recipe recommendations. By leveraging a comprehensive dataset of recipes, we aim to create a tool that not only saves users time but also promotes culinary exploration tailored to their specific needs. This report presents the end-to-end process of building the recommendation system, encompassing data collection, preprocessing, feature engineering, model development, and system evaluation. The dataset utilized in this project is the "Recipe Box" collection from GitHub, which comprises over 125,000 recipes. Key features within the dataset include recipe titles, ingredients, and cooking instructions.

Data Processing

Before building the recipe recommendation system, it was important to process and clean the data to ensure its quality and usability. The dataset, which included around 125,000 recipes, required several steps of preparation to make it suitable for analysis system-building. The following steps were performed:

1. **Ingredient Cleaning:** The first step we took is cleaning the ingredient lists. Many recipes contained unnecessary text, which were not relevant for the recommendation system, so these were removed, and the ingredient lists

were standardized to ensure consistency. This step was crucial to make the ingredient data uniform and easier to work with.

2. **Instruction Cleaning:** Next, the cooking instructions were cleaned to improve their readability and consistency. The text was converted to lowercase, and punctuation marks were removed. Additionally, common stopwords (e.g., "and," "the," "is") were eliminated to reduce noise in the data. This step helped make those instructions more suitable for text-based analysis and processing.
3. **Cook Time Extraction:** Cook time was another important feature that needed standardization. The original data contained cook times in various formats, such as hours and minutes. These were converted into the same format, with all times expressed in minutes. Additionally, to ensure the recommendations were practical and catering to the majority, recipes with extreme cook times (over 3 hours) were removed from the dataset because those kinds of recipes are regarded as complicated recipes, which might not be useful under a limited ingredients scenario. Most of the people do not have the time to cook. This step helped focus on recipes that are more feasible for everyday cooking.
4. **Add Taste Variable:** Finally, to enhance the recommendation system, a new variable called "Taste" was introduced to the original dataset. This variable was created by classifying recipes based on their ingredients. For example, recipes with ingredients like sugar or honey were categorized as "sweet," while those with spices or chili were labeled as "spicy." This addition allowed the system to consider users' taste preferences when making recommendations, making the results more personalized and relevant.

By applying those steps and methods, we processed and cleaned the dataset, ensuring that it was ready for the next steps in building the recommendation system.

Exploratory Data Analysis (EDA)

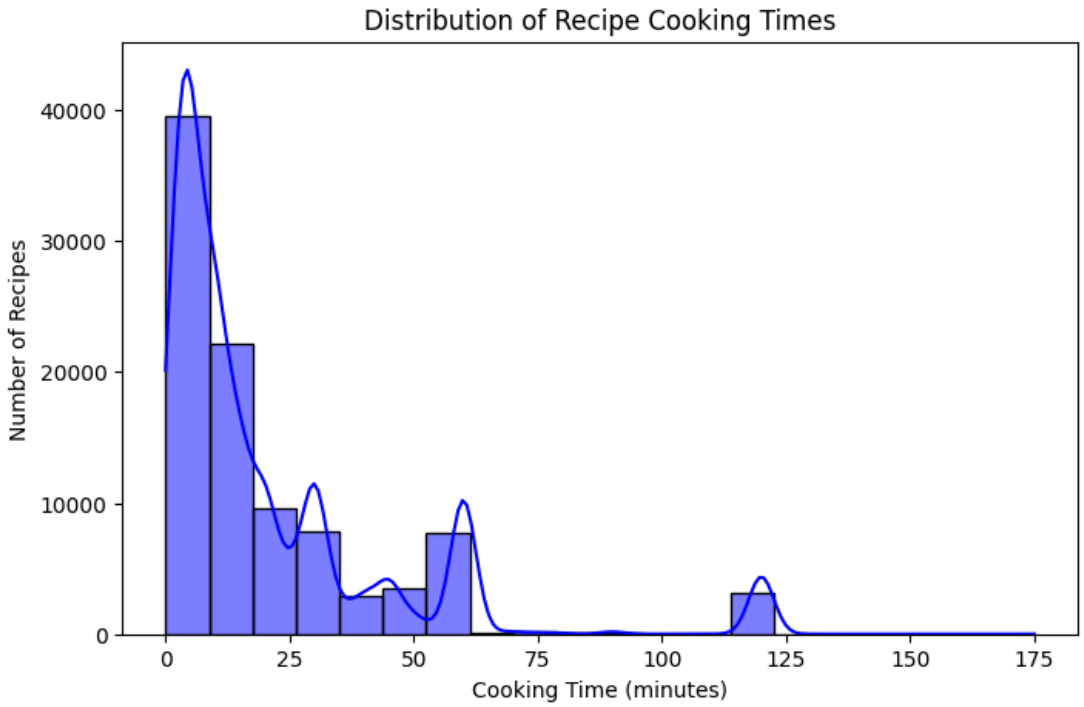

Table 1

Distribution of Recipe Cooking Times

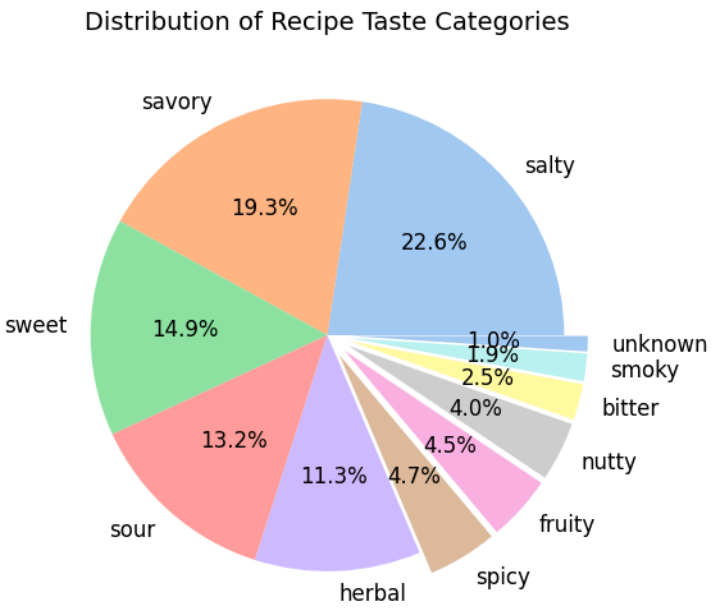

Figure 1 Distribution of Recipe Taste Categories

After we cleaned the dataset, we did the exploratory data analysis (EDA) to help us further dig into the dataset and draw some insights. This analysis focused on two key aspects: the distribution of cooking times and the distribution of taste categories across the recipes. Below are the detailed findings:

1. **Distribution of Recipe Cooking Times**:
   The cooking times of the recipes were analyzed to understand how long most recipes take to prepare. Most recipes fall within a range of 0 to 25 minutes, with a peak of around 10 minutes. This suggests that most recipes in the dataset are designed to be relatively quick and practical for everyday cooking. Recipes with extremely long cooking times (over 125 minutes) were rare, which aligns with our earlier decision to remove recipes with cook times exceeding 3 hours.

2. **Distribution of Recipe Taste Categories**:
   The dataset was also analyzed based on taste categories, which were derived from the ingredients in each recipe. The most common taste category was salty, making up 22.6% of the recipes, followed by savory at 19.3%. Other taste categories, such as sweet, sour, herbal, spicy, and fruity, were less common but still present in significant proportions. A small percentage of recipes (1.0%) were categorized as unknown, likely due to insufficient information in the ingredient lists. This distribution highlights the diversity of tastes in the dataset, allowing the recommendation system to cater to a wide range of user preferences.

In summary, the EDA shows that the dataset is well-suited for building a practical and user-friendly recipe recommendation system. Most recipes have reasonable cooking times, and the variety of taste categories ensures that users can find recipes that match their preferences.

Recommendation Algorithm

To build an effective recipe recommendation system, we designed a step-by-step algorithm that processes user input and matches it with the most suitable recipes from the dataset. Here is a detailed explanation of how the algorithm works:

1. **User Input:**
   The process begins with the user providing their preferences, including the ingredients they have on hand, the maximum cooking time they can allocate, and their desired flavor. This input is the foundation for generating personalized recommendations and each aspect is required.

2. **Data Preprocessing:**
   Before matching the user's input with recipes, the dataset undergoes preprocessing. This includes cleaning the text that the user typed in, extracting cooking times, and classifying recipes based on their desired flavor.

3. **Feature Extraction:**
   To compare the user's input with the recipes, we used the TF-IDF method. This method converts the text into numerical vectors. By doing so, we can mathematically analyze and compare the similarity between the user's input and the recipes in the dataset, which is a key contributor to the generation of recommended recipes.

4. **Calculate Similarity:**
   Once the data is converted into vectors, we use Cosine Similarity to measure how closely the user's input matches each recipe. Cosine Similarity calculates the angle between two vectors, with a higher score indicating a closer match. This step helps identify the recipes that best align with the user's preferences.

5. **Filter & Refine:**
   After calculating similarity scores, the algorithm filters the results to ensure they meet the user's specific requirements. For example, it removes recipes that exceed the user's specified cooking time or require ingredients the user

does not have. This refinement step ensures that the recommendations are practical and relevant.

6. **Output Recommendation Results:**
   Finally, the system returns the top 5 best-matching recipes to the user. These recipes are ranked based on their similarity scores. By limiting the results to the top 5, the system provides a focused set of options, making it easier for the user to decide what to cook.

```
recommendations = recommend_recipes(["chicken", "garlic", "lemon"], max_time=30, preferred_taste="savory")
display(recommendations)
```

	title	ingredients	cookTime	taste	similarity
47332	Lemon Chicken	1 tablespoon butter 1 tablespoon olive oil 2 b...	2.0	[savory, herbal, sour]	0.574074
31045	Home-Style Chicken Piccata	1/2 cup all-purpose flour 2 teaspoons garlic p...	3.0	[savory, sour]	0.533424
81843	Quickest Roasted Chicken Dinner	2 chicken breasts with skin and on the bone 4 ...	25.0	[savory, herbal, sour, salty]	0.524106
18024	Rich Herb and Lemon Chicken	4 skinless, boneless chicken breast halves sal...	2.0	[savory, herbal, sour, salty]	0.518332
45169	Rosemary-Roasted Chicken and Garlic	2 chicken breast halves with skin and bones 2 ...	15.0	[savory, herbal, salty]	0.513002

Figure 2 Python Result Sample

To enhance user experience, we designed a modern, user-friendly interface sample for the Recipe Recommendation System. This sample focuses on clarity, simplicity, and ease of interaction, ensuring users can quickly input their preferences and receive recipe recommendations efficiently.

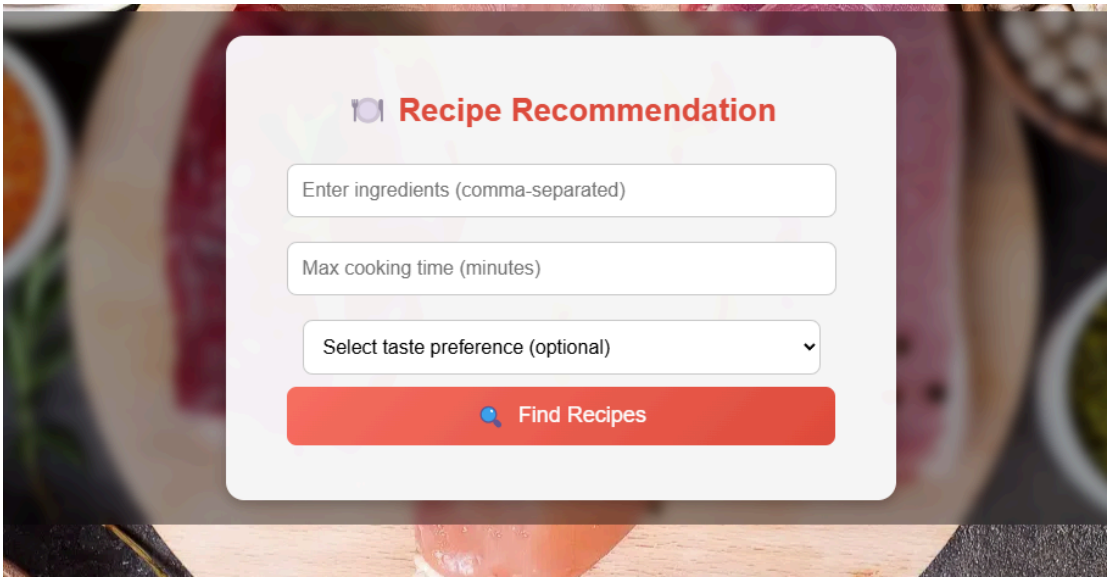

Figure 3 User Interface Sample

Model Evaluation

To assess the effectiveness of our recipe recommendation system, we conducted a series of evaluations based on accuracy, relevance, and ranking of the recommended results. The evaluation metrics used include Accuracy (Simple Relevance Fraction), Precision top 5, and Mean Reciprocal Rank (MRR).

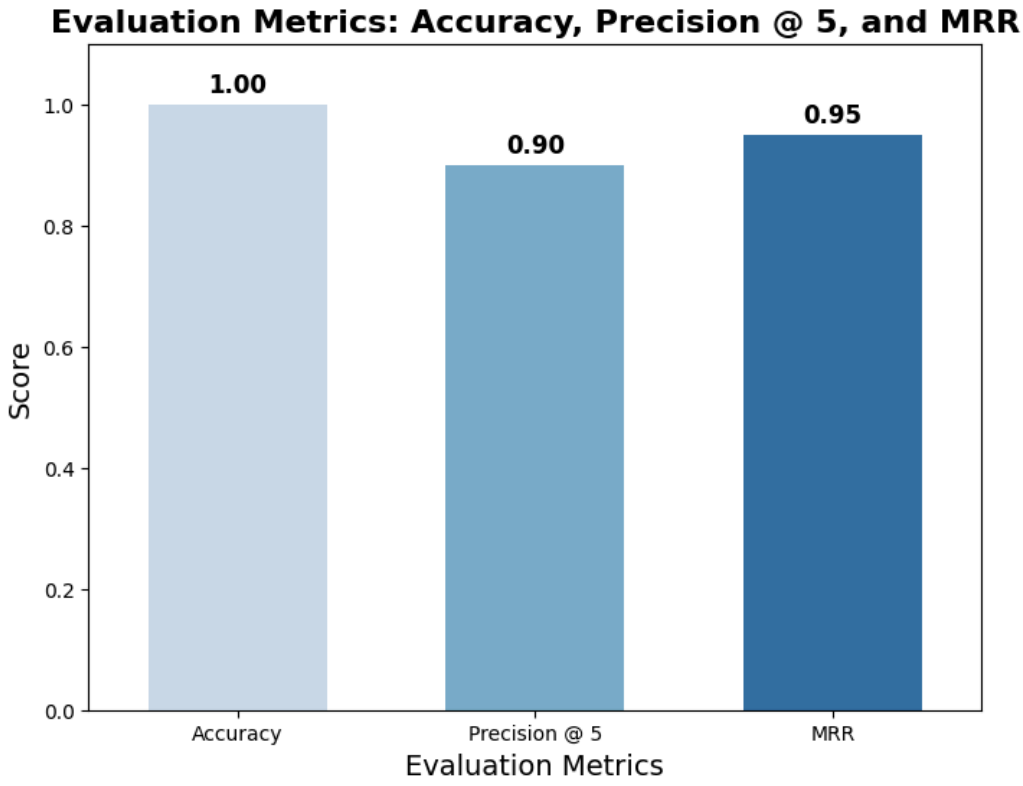

Table 2 Evaluation Metrics Result

- Accuracy (1.00): The model demonstrates high accuracy in recommending relevant recipes that align with user input.
- Precision top 5 (0.90): A significant proportion of the top 5 recommended recipes are accurately matched to user preferences.
- Mean Reciprocal Rank (MRR) (0.95): The most relevant recipes are typically ranked at the top, reducing users' search time and improving the recommendation quality.

These results indicate that the system effectively identifies recipes that fit users' specified ingredients, time constraints, and taste preferences. By ranking the most relevant recipes higher, the system enhances user experience by providing quick and efficient recommendations.

Current Challenges

Despite the strong performance of our recommendation system, there are several limitations that present opportunities for future improvement:

1. **Limited Recommendation Diversity**

   The current system tends to suggest highly similar recipes, which may reduce the diversity of recommendations.

2. **Handling Rare Ingredients**

   When users input less common ingredients, the system may struggle to find suitable matches, leading to suboptimal recommendations.

3. **Data Source Expansion**

   The dataset currently relies on a single fixed source (Recipe Box from GitHub). Integrating multiple recipe databases could enhance coverage and robustness.

Future Improvements

To further enhance the performance and applicability of our system, we propose the following improvements:

1. **Improved Algorithms**

   Integrate content-based filtering with collaborative filtering to improve recommendation diversity and accuracy.

2. **Nutritional Analysis Integration**

   Enhance the system by incorporating calorie and nutrient information, enabling users to receive healthier recipe recommendations.

3. **Multi-Language Support**

   Allow users to input ingredients in multiple languages to make the system more accessible to a global audience.

By addressing these challenges and implementing these improvements, we aim to make the recipe recommendation system more versatile, user-friendly, and widely applicable.

Conclusion

This report presents the development of a content-based recipe recommendation system that helps users quickly find recipes that match their available ingredients, time constraints, and taste preferences. The system leverages a large recipe dataset, cleans and preprocesses the data, and applies TF-IDF and Cosine Similarity to generate relevant recommendations.

Our evaluation demonstrates high accuracy (1.00), strong precision (0.90), and effective ranking (MRR 0.95), confirming the system's ability to provide personalized and efficient recipe suggestions. While challenges remain—such as recommendation diversity and handling rare ingredients—future improvements, including algorithm enhancements, nutritional analysis, and multilingual support, will further refine the system.

By reducing the time spent searching for recipes and maximizing ingredient utilization, our system provides a practical and user-friendly solution for everyday cooking, making meal preparation more convenient and enjoyable.