Integrating Convolutional Neural Networks and Meta LLaMA2 for Recipe Generation from Ingredient Lists and Images

Aparna R. Sawant, Soham R. Karandikar, Prasad A. Kamble, Kshitij S. Kalrao, Parth V. Kallurwar, Jaee R. Kale

Department of Information Technology, Vishwakarma Institute of Technology, Pune, India

Abstract

This paper proposes a system that integrates Convolutional Neural Networks (CNNs) and the Meta LLaMA2 language model to generate structured recipes from either textual ingredient lists or food images. CNNs identify ingredients from images, while the fine-tuned LLaMA2 model (trained on the RecipeNLG dataset) generates coherent recipes. A real-time, interactive interface is built using Streamlit and Flask in a client-server setup. Evaluation using BLEU and ROUGE-L metrics validates the model's effectiveness. The work contributes to intelligent and personalised recipe recommendation systems with real-world applicability.

Keywords: recipe generation, CNN, Meta LLaMA2, image classification, language models.

1 Introduction

Technological advancements have reshaped everyday life, including cooking practices [1]. With the popularity of online recipe platforms, there is increasing interest in systems that can generate recipes using AI and available ingredients [2]. This study explores the integration of Convolutional Neural Networks (CNNs) and the Meta LLaMA2 language model to develop a sys-

tem that generates recipes from either a list of ingredients or corresponding images. CNNs handle image-based ingredient recognition, while LLaMA2 excels in contextual natural language generation [3, 4]. The system aims to enhance culinary accessibility, encourage creativity, and enable personalised meal preparation. It transforms conventional kitchens into spaces of intelligent automation and gastronomic exploration.

2 Related Work

Significant advancements have been made in ingredient recognition and recipe generation using Convolutional Neural Networks (CNNs) and large language models (LLMs). Touvron et al. [5] introduced Meta LLaMA2, which was made publicly accessible via Huggingface [6], facilitating model fine-tuning and deployment. Practical strategies for adapting LLMs using the Transformers and Datasets libraries have been discussed in [7, 8]. RecipeNLG [9], derived from Recipe1M+ [10], serves as a foundational dataset in culinary text generation. In the domain of ingredient recognition, Rodrigues et al. [11] and Morol et al. [12] proposed CNN-based systems using datasets such as Fruits and Vegetables Recognition. Transfer learning techniques, including VGG and MobileNet, have been applied to fruit classification and quality detection in [13], consistently achieving robust results. These studies collectively motivate the integration of CNNs for image-based recognition and LLMs for recipe generation in the proposed system.

3 Proposed Methodology

3.1 Block Diagram

The high-level architecture of the proposed system is shown in Figure 1. It is structured into two primary components: the client and the server. The client includes the user interface and a lightweight Convolutional Neural Network (CNN) for ingredient recognition. The server, hosted on Google Colaboratory using Flask and accessed via Ngrok, runs the Meta LLaMA2-based recipe generation model. Users can either input a list of ingredients or upload an image, which the CNN processes to identify ingredients. This list is sent to the server, which generates a unique identifier (UUID) and launches a background thread to run LLaMA2 for token-by-token recipe generation. These tokens are streamed to a Firestore database under the associated UUID.

The client polls the database every two seconds to fetch new tokens, allowing the user to view the recipe incrementally in real time. This streaming approach enhances interactivity without waiting for the full 1–2 minute generation cycle.

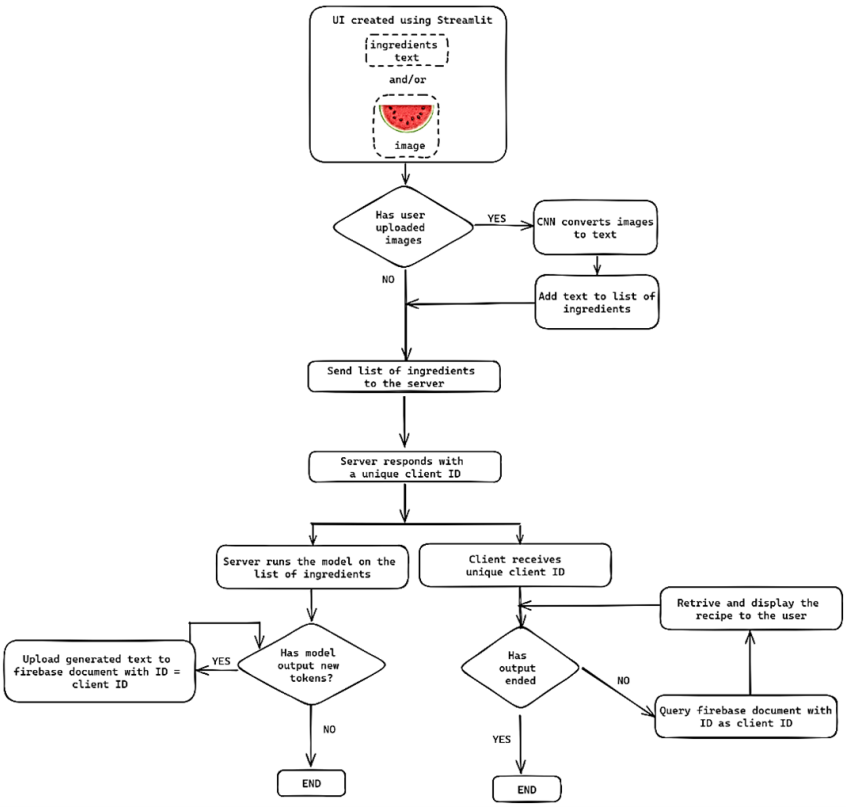


Figure 1 High-level block diagram of the proposed system architecture

3.2 Creating the CNN for Ingredient Identification

CNN development began with dataset loading and preprocessing using `tf.keras.utils.image_dataset_from_directory()`, resizing images to 64×64 in RGB format. This ensured consistency for training. The CNN architecture, defined in TensorFlow, comprised convolutional layers for feature extraction, pooling layers for downsampling, and fully connected layers for classification.

It was designed to balance performance and computational efficiency. The model was compiled with a suitable optimiser and loss function, using accuracy as the primary evaluation metric. Training was conducted using the `fit()` method with batches of 32 samples, across multiple epochs. Validation metrics were monitored to assess generalisation and detect overfitting. After achieving satisfactory performance, the model was saved in `.h5` format as `trained_model.h5`, making it ready for deployment without requiring retraining.

3.3 Fine-tuning Meta LLaMA2 for Recipe Generation

The Meta LLaMA2 model was fine-tuned using the RecipeNLG dataset containing approximately two million recipes, accessed and preprocessed via the Huggingface Datasets library. Ingredient names, often embedded with quantities, and newline-separated instructions were cleaned and reformatted using FoodBERT to suit prompt templates aligned with LLaMA2 documentation. The prompt-structured dataset was uploaded to the Huggingface Hub and fine-tuned on Google Colaboratory using an NVIDIA T4 GPU. Among several LLaMA2-7B checkpoints, the chat-optimised variant was selected for its superior output quality. Memory constraints were addressed by sharding across CPU, GPU, and disk, and fine-tuning was implemented using QLoRA (Quantised Low-Rank Adapters), enabling 4-bit weight quantisation for memory efficiency. The training loop, managed via the `SFTTrainer` class from Accelerate and BitsAndBytes, used a learning rate of $1e$–5, LoRA attention size of 64, LoRA alpha of 16, and a dropout rate of 0.1. Training was limited to one epoch and 1000 steps, after which adapter weights were saved and published to Huggingface Hub. The same configuration was reused to fine-tune GPT-2 and Mistral 7B Instruct models for comparative analysis.

3.4 Deployment

The trained models were integrated into a web-based system combining Streamlit for the client and Flask for the server. The CNN model (`trained_model.h5`) was deployed client-side using Streamlit to process user-inputted text or uploaded images for ingredient recognition. On the server side, a Flask application hosted on Google Colaboratory (exposed via Ngrok) received the processed ingredient list, generated a UUID, and initiated inference using the fine-tuned LLaMA2 model. Generated recipe tokens were incrementally written to a Firestore database under the corresponding UUID.

The client polled the database every two seconds, enabling real-time recipe display. This deployment ensured smooth integration between client-side recognition and server-side generation, demonstrating the practical use of large language models in resource-constrained environments while offering a responsive and user-friendly interface.

4 Results and Discussion

4.1 Ingredient Identification Results

The CNN model demonstrated consistent accuracy in identifying food ingredients from images. For instance, as shown in Figure 2, a sample beetroot image was correctly classified, confirming the model's generalisation on unseen data. The Streamlit interface enabled real-time user interaction and immediate prediction feedback.

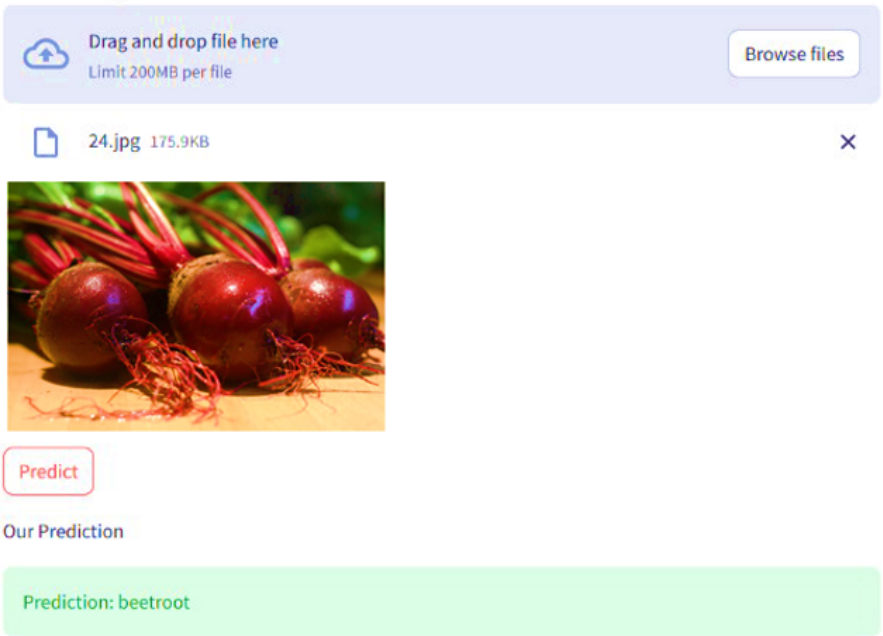
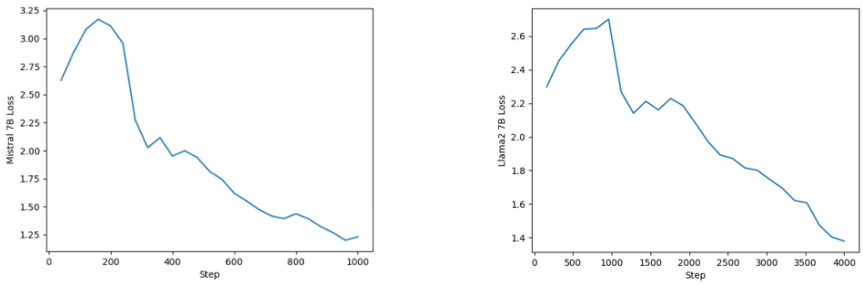


Figure 2 Prediction of ingredient from image using CNN model (Classified as beetroot)

4.2 Training Loss Analysis

Figure 3 shows the loss curves for LLaMA2 7B and Mistral 7B. LLaMA2's loss reduced steadily from 2.6 to below 1.4 over 4000 steps, indicating stable convergence. Mistral, though trained for fewer steps, showed a sharper drop from 3.2 to 1.25. Both models exhibited successful fine-tuning, with LLaMA2 achieving smoother convergence over a longer duration.



(a) LLaMA2 7B model (b) Mistral 7B model

Figure 3 Training loss vs. number of steps for LLaMA2 and Mistral 7B models

4.3 Evaluation Metrics

Model performance was assessed using BLEU, Brevity Penalty, and ROUGE-L scores, as summarised in Table 1. LLaMA2 7B achieved a BLEU score of 0.19 and ROUGE-L of 0.28, outperforming Mistral 7B. While GPT-2 showed a higher BLEU score, it incurred a brevity penalty due to its shorter outputs.

Table 1 Model performance based on BLEU, Brevity Penalty, and ROUGE-L

Model	**BLEU**	**Brevity Penalty**	**ROUGE-L**
LLaMA2 7B	0.19	1.0	0.28
Mistralv0.2 Instruct 7B	0.13	1.0	0.21
GPT-2 335M(Lee et al., 2020)	8.85	0.71	0.37

4.4 Recipe Generation Interface

The Streamlit interface enabled real-time recipe generation from both text and image inputs. As shown in Figure 4, a sample output for a watermelon and tomato salad demonstrates dynamic display of generated tokens retrieved via UUID from Firestore. This progressive rendering enhanced interactivity by eliminating long wait times.

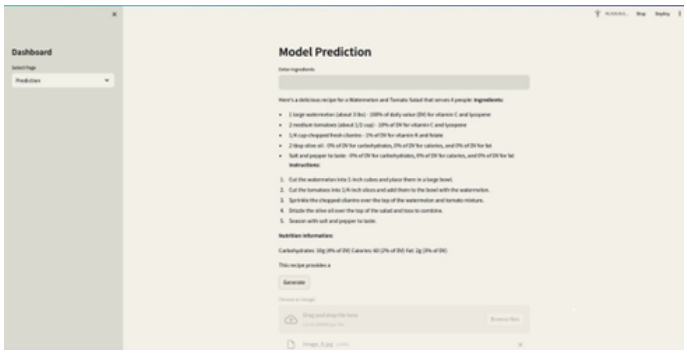


Figure 4 Recipe generation interface showing a watermelon, tomato, and coriander salad recipe

5 Conclusion

This study presents an integrated system for automated recipe generation by combining CNN-based ingredient recognition with natural language generation using a fine-tuned Meta LLaMA2 model. The system supports input from both text and food images, providing real-time, interactive recipe generation through a Streamlit-Flask deployment pipeline. Parameter-efficient fine-tuning via QLoRA enabled the model to perform effectively within hardware-constrained environments. Evaluation metrics such as BLEU and ROUGE-L validated the quality of the generated recipes, with the LLaMA2 model outperforming GPT-2 and Mistral variants. The interface offered smooth interaction and immediate feedback, highlighting the system's potential for practical use in culinary AI applications. Looking ahead, future enhancements include extending training over multiple epochs on high-memory GPUs and introducing validation sets to improve generalization. Incorporating culturally diverse datasets—particularly traditional Indian recipes annotated with spice profiles and cooking styles—can enhance localization.

Technically, alternative fine-tuning approaches like Representation Fine-Tuning (ReFT) may further boost efficiency. Additional features such as dietary filters, ingredient substitutions, and reinforcement learning-based user feedback could enhance personalization and health relevance. Overall, the proposed system sets the groundwork for intelligent, culturally adaptive, and user-aware digital kitchen assistants and recipe recommendation tools.

References

[1] J.M. Pilcher. *Food in World History*. Routledge, 2023.
[2] M. Sadhale. The concept of home cooks and its entrepreneurial business potential during and post COVID-19 pandemic. *International Journal of Future Generation Communication and Networking*, 14(1):1752–1772, 2021.
[3] P. Ma, Y. Li, M. Xu, and X. Liu. Large language models in food science: Innovations, applications, and future. *Trends in Food Science & Technology*, 2024.
[4] X. Tu, Y. Ma, Q. Huang, and X. Zhang. An overview of large AI models and their applications. *Visual Intelligence*, 2(1):1–22, 2024.
[5] H. Touvron, L. Martin, K. Stone, et al. LLaMA 2: Open foundation and fine-tuned chat models. Technical Report, Meta AI, 2023.
[6] M.A.K. Raiaan, M.S. Hossain, A. Shahjalal, and M.M. Hassan. A review on large language models: Architectures, applications, taxonomies, open issues and challenges. *IEEE Access*, 12:26839–26874, 2024.
[7] H. Naveed, T. Ahmad, A. Iqbal, and A. Majeed. A comprehensive overview of large language models. *arXiv preprint arXiv:2307.06435*, 2023.
[8] T. Wolf, L. Debut, V. Sanh, et al. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, 2020.
[9] M. Bień, P. Suwała, and K. Kajdanowicz. RecipeNLG: A cooking recipes dataset for semi-structured text generation. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 22–28, 2020.
[10] J. Marín, H. Wang, N. Ahuja, and A. Torralba. Recipe1M+: A dataset for learning cross-modal embeddings for cooking recipes and food images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(1):187–203, 2021.
[11] M.S. Rodrigues, J.R. Oliveira, and F.J. Soares. RecipeIS—Recipe recommendation system based on recognition of food ingredients. *Applied Sciences*, 13(13):7880, 2023.
[12] M.K. Morol, M.M. Hossain, M.T. Hasan, and M.H. Kabir. Food recipe recommendation based on ingredients detection using deep learning. In *Proceedings of the 2nd International Conference on Computing Advancements*, pages 191–198, 2022.
[13] H. Muresan and M. Oltean. Fruit recognition from images using deep learning. *Acta Universitatis Sapientiae, Informatica*, 10(1):26–42, 2018.

Biography

Aparna R. Sawant is an Assistant Professor in the Department of Information Technology at Vishwakarma Institute of Technology, Pune. Her research interests include Natural Language Processing and intelligent systems.

Soham R. Karandikar is a third-year B.Tech. student at Vishwakarma Institute of Technology, Pune. He is passionate about full-stack development and has experience in fine-tuning large language models and operating systems.

Prasad A. Kamble is a third-year B.Tech. student at Vishwakarma Institute of Technology, Pune, with interests in Artificial Intelligence, Image Processing, and Data Science. He is skilled in Java, R, and C.

Kshitij S. Kalrao is pursuing a B.Tech. in Information Technology at Vishwakarma Institute of Technology, Pune. He has worked on projects in Machine Learning, Deep Learning, IoT, and Embedded Systems.

Parth V. Kallurwar is an undergraduate student in Information Technology at Vishwakarma Institute of Technology, Pune. He is enthusiastic about exploring new technologies and expanding his technical knowledge.

Jaee R. Kale is a third-year B.Tech. student at Vishwakarma Institute of Technology, Pune. She has worked on projects involving Flask, YOLO, and Redis, with interests in Machine Learning and Image Processing.