Pixel to Plate

Student Name: Ieshaan Awasthy
Roll Number: 2021054

Student Name: Gunjan Dabas
Roll Number: 2021253

BTP report submitted in partial fulfillment of the requirements for the Degree of B.Tech. in Computer Science & Engineering (Ieshaan) and Computer Science and Applied Mathematics (Gunjan)

Date: 13th December 2024

BTP Track: Engineering Track

BTP Advisor: Dr. Ganesh Bagler

**Indraprastha Institute of Information Technology
New Delhi**

Student’s Declaration

I hereby declare that the work presented in the report entitled **“Pixel To Plate”** submitted by me for the partial fulfillment of the requirements for the degree of *Bachelor of Technology* in *Computer Science & Engineering (Ieshaan)* and *Computer Science and Applied Mathematics (Gunjan)* at Indraprastha Institute of Information Technology, Delhi, is an authentic record of my work carried out under guidance of **Dr.Ganesh Bagler**. Due acknowledgements have been given in the report to all material used. This work has not been submitted anywhere else for the reward of any other degree.

Ieshaan Awasthy and Gunjan Dabas
(student’s name)

Place & Date: 28/11/2024

Certificate

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Dr.Ganesh Bagler
(advisors’ name)

Place & Date: ....

Abstract

This project, *Pixel to Plate*, aims to bridge computer vision and natural language processing to automate recipe generation from images of ingredients. The first phase of the project focuses on object detection, employing state-of-the-art YOLO models to accurately identify ingredients in the AI-Cook dataset. Comprehensive exploratory data analysis (EDA) was conducted to address dataset quality, class imbalance, and object co-occurrence patterns. Among the tested models, YOLOv8x demonstrated superior performance with a precision of 0.970, making it the chosen model for ingredient detection.

The second phase evaluates four large language models (LLaMA, Falcon, GEMMA, and Phi) for recipe generation based on detected ingredients. Models were assessed in a zero-shot setting for coherence, completeness, and relevance. The analysis revealed that LLaMA outperformed the others, producing recipes with logical structure, meaningful use of ingredients, and balanced food combinations.

This interdisciplinary effort highlights the potential of combining advanced computer vision and language models for culinary applications, paving the way for automated recipe generation systems that could transform personalized cooking experiences. The findings underscore the importance of model selection, data quality, and task-specific evaluation metrics in achieving reliable results.

Keywords: (Object Detection, YOLO, Recipe Generation, Large Language Models, AI-Cook Dataset, Natural Language Processing, Computer Vision, Culinary Applications)

Acknowledgments

We would like to express our deepest gratitude to our BTP advisor, Dr. Ganesh Baglar, for his invaluable guidance, support, and encouragement throughout the duration of this project. His expertise and insightful feedback were instrumental in shaping our approach and helping us achieve our goals.

We are also extremely thankful to PhD Mansi Goel for her continuous support and encouragement, particularly in the areas of computer vision and natural language processing. Her knowledge and insights were crucial in refining our model designs and understanding the technical intricacies of this project.

Additionally, we would like to thank Madhavi Ma'am for her support and guidance throughout the project, helping us overcome challenges and ensuring we stayed on track towards completing this BTP successfully.

Finally, we extend our appreciation to the entire faculty of the Computer Science and Engineering Department at Indraprastha Institute of Information Technology, Delhi, for their constant support, encouragement, and academic resources throughout this research endeavor.

Work Distribution

The work distribution between us was equal, with both contributors equally involved in all aspects of the project. Both Ieshaan Awasthy and Gunjan Dabas contributed to the object detection phase, model evaluation, and analysis of the results. The recipe generation phase was also a collaborative effort, with both working on the evaluation of different large language models and their performance metrics.

We would like to extend our gratitude to the entire faculty and staff of the Indraprastha Institute of Information Technology, New Delhi, for their support and encouragement.

Contents

Chapter 1

Introduction

1.1 Motivation

The motivation for this project stems from the need to integrate distinct AI capabilities, such as image recognition and natural language generation, to solve real-world problems. While advancements in these areas have been significant, their combined potential for creating practical, user-centric solutions remains underutilized. This project addresses this gap by tackling a common problem: generating recipes from the ingredients available in a refrigerator.

Determining what to cook often requires creativity, dietary considerations, and careful planning. Automating this task simplifies meal preparation, making it more efficient and accessible. By generating personalized, context-aware recipes, the system enhances convenience while addressing individual preferences such as dietary restrictions, cost, and preparation time.

The project's modular design is another key motivation. Unlike monolithic systems, a modular approach enables independent optimization and scalability. Each component evolves independently, allowing seamless integration of improvements in object detection or language generation technologies. This flexibility ensures adaptability and long-term relevance.

Beyond its immediate application, the project demonstrates how AI can be tailored to niche domains, offering a blueprint for similar efforts in areas like healthcare or education. By integrating vision and language capabilities, this work sets a precedent for creating intelligent systems that enhance daily life.

1.2 Problem Statement

Households often need help deciding what to cook using available ingredients, requiring effort to balance dietary restrictions, budgets, and preparation time. Existing tools lack adaptability and fail to provide personalized solutions. The challenge is to create a system that identifies ingredients from images and generates context-aware recipes. This requires integrating advanced object detection with language generation in a modular, scalable framework to deliver practical and user-centric solutions.

Chapter 2

Research on Existing Work

The AI community has adopted multisensory or multimodal techniques to enhance the current generation of AI models, with the aim of achieving more comprehensive intelligent understanding. Integrating language and imagery is a well-established approach for tasks such as image captioning or generating visuals from descriptions. The Multimodal And Modular Ai Chef: Complex Recipe Generation From Imagery, addresses a growing need in AI applications to combine multimodal capabilities—specifically image recognition and language generation—to solve practical problems. The authors propose a modular framework, which effectively separates the tasks of object detection and recipe generation. Using YOLOv5 for object detection and OpenAI's GPT-3.5 for recipe generation, the system demonstrates how combining specialized models can outperform monolithic multimodal approaches in specific use cases.

2.1 Introduction

- Multimodal learning, which integrates sensory inputs like text and imagery, has become a cornerstone of AI development.
- Existing models like CLIP, GATO, and LAION-5b exhibit capabilities in image-to-text tasks but lack coherence in maintaining context for complex applications like recipe generation.
- The study explores a modular alternative where image detection models identify objects and pass them to a Large Language Model (LLM) for text generation.
- The main goal is to create recipes tailored to constraints such as dietary restrictions, preparation time, cost, and portion sizes.

2.2 Methodology

2.2.1 1. Image Detection

- Utilized YOLOv5 (small) as the object detection model, trained on the *ai-cook-lcv4d* dataset, which includes 3,050 images of 30 common refrigerator items.
- Training involved a split of 2,896 images for training, 103 for validation, and 51 for testing.

- Pre-processing included augmentations such as rotation, exposure adjustments, and noise addition.
- Achieved a mean average precision (mAP) of 95.2% and high recall rates for ingredient detection.

2.2.2 2. Recipe Generation

- Used OpenAI's GPT-4 model for text generation.
- The input was a delimited list of detected ingredients, and the output included:
  - Recipe title.
  - Ingredients list with quantities.
  - Cooking instructions with step-by-step details.
  - Approximate preparation and cooking times.
- Recipes adhered strictly to the provided ingredient list without external supplementation.
- Variants were generated for specific constraints such as vegan, keto, or lactose-intolerant diets.

2.2.3 3. Modular API Pipeline

- The modular approach separates the image detection and text generation stages, enabling independent updates to each component.
- This design reduces computational overhead and allows rapid adaptation to evolving models.
- The system can handle scenarios like ingredient expiration tracking, meal planning, and waste reduction.

2.3 Results

- Generated a 100-page recipe book featuring recipes based on 30 primary ingredients derived from 2,000 refrigerator images.
- Achieved high accuracy in ingredient detection, as reflected in the multi-class confusion matrix.
- Demonstrated versatility in generating diverse recipe variations:
  - Adjusting for dietary preferences (vegan, keto, lactose-free).
  - Optimizing for cost-efficiency and seasonal availability.
  - Customizing portion sizes and meal types.
- Showcased the ability to refine recipes interactively based on user feedback, leveraging GPT-4's long conversational memory.

2.4 Discussion

- The modular design supports practical applications by optimizing resource use and enabling real-time edge deployment.
- Potential future enhancements include:
  - Incorporating reinforcement learning to improve recipe generation based on user feedback.
  - Expanding to include global cuisines and advanced food pairing principles.
  - Adding operational research capabilities for cost optimization in large-scale settings like restaurants.
- Highlights the environmental benefits of reducing food waste through efficient inventory utilization.

2.5 Conclusion

The study demonstrates a scalable, practical solution for AI-based recipe generation by combining state-of-the-art image detection and language generation models. The modular API approach outperforms monolithic models in adaptability and efficiency, enabling complex human-like culinary tasks in real-world scenarios.

Chapter 3

Work Done

3.1 Object Detection

The primary focus of this phase was to identify the most suitable object detection model for the AI-Cook dataset. This model will be used to detect ingredients and subsequently facilitate automated recipe generation. Below are the detailed steps and outcomes of this phase:

3.1.1 Dataset Preparation

The dataset was sourced from Roboflow and thoroughly analyzed to ensure its readiness for model training. Key aspects of the dataset are as follows:

- **Structure:** The dataset consists of 3,040 images with corresponding label files, organized into training (2,896 images), validation (93 images), and test sets (51 images).
- **Image Properties:** All images have uniform dimensions of 640x640 pixels, ensuring consistency during training.
- **Class Distribution:** Class imbalance was noted, with some categories underrepresented. Frequent co-occurrence of specific class pairs (e.g., bread & butter) provides valuable insights for future tasks.
- **Bounding Boxes:** Bounding box sizes varied, with most being relatively small compared to the image size. This was confirmed by analyzing the bounding box-to-image size ratio.
- **Object Density:** Most images contain multiple objects, which enhances the dataset's representativeness for real-world scenarios.

3.1.2 Exploratory Data Analysis (EDA)

Detailed EDA was conducted to address potential issues such as class imbalance and data quality. Highlights include:

- No duplicate or blurry images were found in the dataset.
- Analysis of object co-occurrence and density provided insights into natural groupings of ingredients.

3.1.3 Outline of Object Detection Models

To identify the optimal object detection model for ingredient recognition in refrigerator images, we evaluated five YOLO (You Only Look Once) variants. These models represent incremental improvements in object detection technology and were trained under identical conditions to ensure a fair comparison.

Overview of YOLO Variants

YOLO models are renowned for their efficiency and real-time performance in object detection tasks. Below, we provide detailed insights into the architectures and advancements of the variants evaluated:

1. YOLOv5 (s and x Variants)

- YOLOv5 introduced several architectural improvements over YOLOv4, including:
  - Use of the CSPDarknet53 backbone for enhanced feature extraction.
  - Mosaic data augmentation to improve model generalization.
  - An improved anchor-based detection mechanism for precise bounding box localization.
- The `s` (small) variant is designed for computational efficiency, using fewer parameters, making it suitable for edge devices.
- The `x` (extra-large) variant increases the number of layers and parameters, prioritizing accuracy over speed.
- Both variants demonstrated reliable performance but struggled to match the accuracy of more recent models like YOLOv8.

2. YOLOv7

- YOLOv7 builds upon YOLOv4 and introduces Extended Efficient Layer Aggregation Networks (E-ELAN), which:
  - Enhance the learning capacity of the model without significantly increasing computational overhead.
  - Allow better utilization of feature maps for complex object detection tasks.
- This version achieves a superior balance between inference speed and detection accuracy compared to YOLOv5.
- YOLOv7 performs exceptionally well in scenarios requiring high precision, such as identifying small or overlapping objects in cluttered environments like refrigerators.

3. YOLOv8 (s and x Variants)

- YOLOv8 introduces cutting-edge improvements, including:
  - An updated backbone network optimized for feature representation.
  - Anchor-free detection for improved performance on irregularly shaped objects.
  - Enhanced feature pyramid network (FPN) and path aggregation network (PAN) for better multi-scale detection.
- The `s` (small) variant remains resource-efficient, suitable for mobile and edge devices.
- The `x` (extra-large) variant offers state-of-the-art accuracy by leveraging deeper and wider architectures, excelling in highly detailed object detection tasks.

3.1.4 Training Details

- All models were trained on the `ai-cook-lcv4d` dataset, consisting of 3,050 images across 30 ingredient categories.
- Images were pre-processed with augmentations including rotation, exposure adjustment, noise addition, and cut-outs to improve generalization.
- The dataset was split into training, validation, and testing sets in a ratio of 56:2:1.
- Training was conducted for 90 epochs on a single NVIDIA A-100 GPU, ensuring sufficient exposure to the dataset for learning.

3.1.5 Evaluation Metrics

To assess model performance, the following metrics were employed:

- **F1 Score:** Evaluates the balance between precision (correctly identified objects) and recall (completeness of detection). A higher F1 score indicates fewer false positives and false negatives.
- **Precision-Recall Curves:** Visualize the trade-offs between precision and recall at various confidence thresholds, providing insights into model behavior under different scenarios.
- **mAP@0.5:** Measures mean average precision at an IoU threshold of 0.5, quantifying the model's accuracy in detecting objects.

3.1.6 Results

Model	F1 Score (Confidence Threshold)	mAP@0.5
YOLOv5s	0.95 at 0.702	0.964
YOLOv5x	0.95 at 0.615	0.961
YOLOv7	0.96 at 0.710	0.955
YOLOv8s	0.96 at 0.709	0.968
YOLOv8x	**0.97 at 0.753**	**0.970**

Table 3.1: Performance Metrics of YOLO Models.

3.1.7 Insights and Key Observations

- **YOLOv5:** While delivering reliable performance, both s and x variants fell short in terms of precision-recall balance compared to newer models.
- **YOLOv7:** Improved learning capacity and feature aggregation enabled higher F1 scores than YOLOv5, but its mAP@0.5 lagged behind YOLOv8.
- **YOLOv8:**
  - YOLOv8s provided a strong balance of speed and accuracy, ideal for resource-constrained applications.
  - YOLOv8x achieved the best performance with an F1 score of 0.97 and mAP@0.5 of 0.970, excelling in all tested metrics.

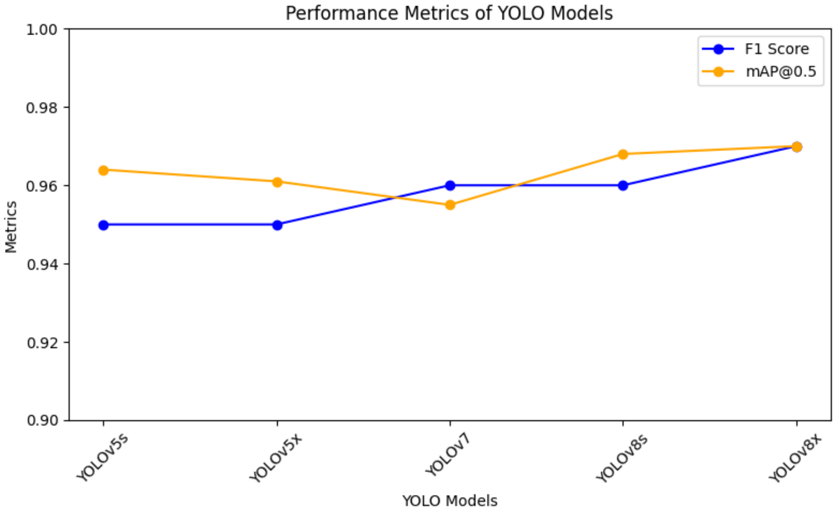
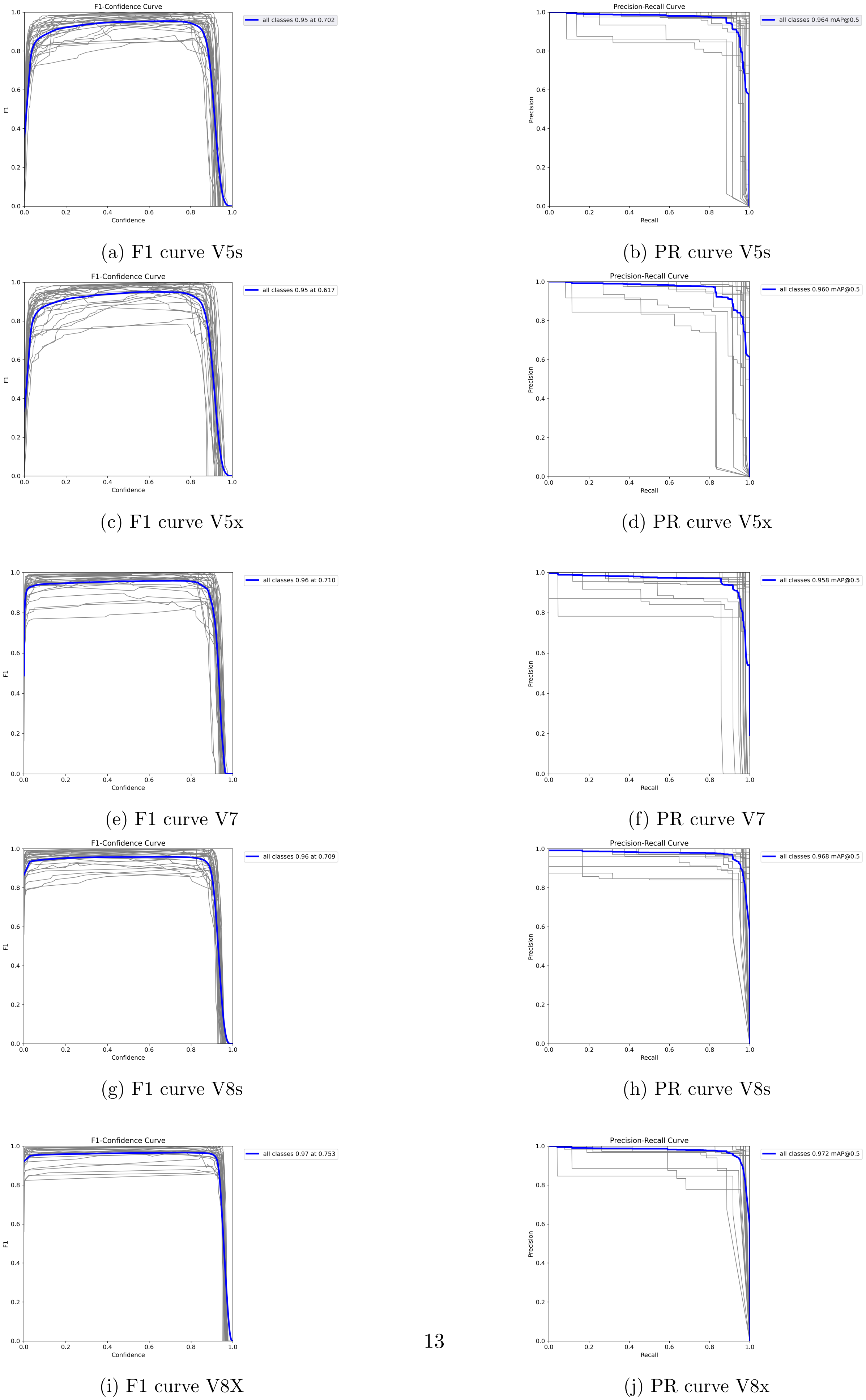


Figure 3.2: YOLO Models' Performance Metrics.

3.1.8 Conclusion

YOLOv8x was selected as the final model for ingredient detection due to its superior performance across all evaluation metrics. Its advanced architecture ensures reliable detection even in challenging conditions, providing a robust foundation for subsequent recipe generation tasks. The modular nature of this approach facilitates future scalability and adaptability as object detection technology evolves..



(a) F1 curve V5s

(b) PR curve V5s

(c) F1 curve V5x

(d) PR curve V5x

(e) F1 curve V7

(f) PR curve V7

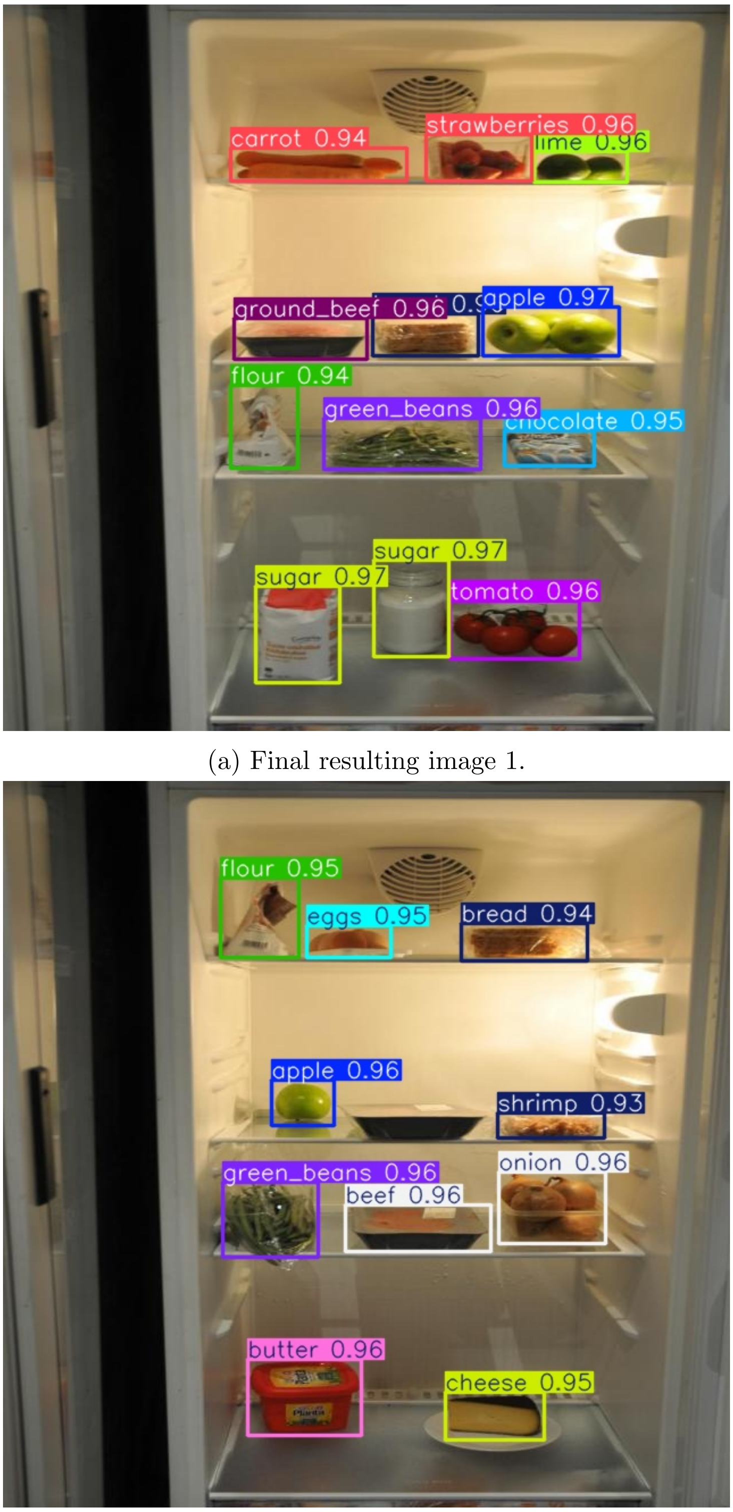
(g) F1 curve V8s

(h) PR curve V8s

(i) F1 curve V8X

(j) PR curve V8x

Figure 3.1: F1 CURVE AND PR CURVE OF MODELS



(a) Final resulting image 1.

(b) Final resulting image 2.

Figure 3.3: Final resulting images from YOLOv8 Model.

3.2 Text Generation

3.2.1 Overview

In this project, we evaluated four state-of-the-art large language models (LLMs) for their ability to generate recipes in a zero-shot setting. The goal was to assess the quality of generated recipes in terms of relevance, coherence, and completeness without prior task-specific fine-tuning. The models tested were:

- Llama-3-8B-Instruct
- Falcon
- Gemma-2B-it
- Phi-3.5-mini-instruct

3.2.2 Brief On LLM used

1. Llama-3-8B-Instruct

About the Model: Llama-3-8B-Instruct, developed by Meta AI, is a transformer-based model optimized for natural language understanding and generation tasks. It offers high efficiency with fewer parameters compared to larger models, and its instruction-based fine-tuning makes it effective at following prompts. Despite its smaller size, it delivers robust performance for a wide range of tasks, including recipe generation.

Why We Included It for Recipe Generation: Llama-3-8B-Instruct is included for its zero-shot capabilities, allowing it to generate coherent recipes from minimal input. It can creatively combine ingredients and generate structured cooking instructions, making it ideal for diverse recipe generation tasks without the need for task-specific training.

2. Falcon

About the Model: Falcon is an open-source transformer-based model designed for high-performance NLP tasks. It excels at generating fluent, human-like text due to its large model size and extensive training data. Its architecture allows for deep understanding and generation of complex, contextually nuanced text.

Why We Included It for Recipe Generation: Falcon is included for its ability to handle complex ingredient lists and cooking methods. It can generate structured recipes that adapt to various culinary styles and dietary needs, making it highly versatile for recipe creation with logical flow and fluency.

3. Gemma-2B-it

About the Model: Gemma-2B-it is a specialized language model fine-tuned for creative tasks. It excels in generating long-form content while maintaining fluency and coherence. Its focus on creative writing makes it suitable for generating detailed recipes with structured steps.

Why We Included It for Recipe Generation: Gemma-2B-it is included for its ability to create clear, detailed recipes based on ingredient lists. Its creativity and coherence help in producing novel recipes with easy-to-follow instructions, making it ideal for personalized recipe generation with specific dietary preferences.

4. Phi-3.5-mini-instruct

About the Model: Phi-3.5-mini-instruct is a recent addition with high contextual understanding and natural language generation capabilities. Built on transformer architecture, it performs well across tasks like text generation and summarization, maintaining coherence and logical flow.

Why We Included It for Recipe Generation: Phi-3.5-mini-instruct is included for its ability to maintain a clear, logical sequence in recipe generation. Its zero-shot capabilities and understanding of both ingredients and cooking steps ensure that it generates coherent and realistic recipes, making it an excellent choice for this project.

3.2.3 Methodology

- **Dataset:** Recipes were generated based on a predefined set of ingredients derived from object detection outputs.
- **Evaluation Metrics:**
  - **Coherence:** Logical consistency and clarity in instructions.
  - **Completeness:** Coverage of all supplied ingredients and recipe steps.
  - **Relevance:** Adherence to the input prompt and task requirements.
- **Testing Setup:** Recipes were generated in a zero-shot context, meaning the models had no fine-tuning for culinary tasks.

3.2.4 Analysis

Clarity

Clarity involves evaluating how well the recipe instructions are structured, whether they are easy to follow, and if the preparation steps are clearly defined.

- **Falcon:** Relatively straightforward but occasionally lack detail in preparation steps, making some processes ambiguous.
- **LLaMA Model Recipes:** Offers a clear structure with detailed instructions. Steps are logically ordered and easy to follow.
- **GEMMA-2B-IT Recipes:** Some recipes are clear, but others have complex instructions or overly condensed steps that might confuse inexperienced cooks.
- **Phi:** Similar to Falcon, with variability in clarity. Some recipes are clearer than others, suggesting inconsistency.

Meaningfulness

This parameter evaluates how purposefully the recipes use ingredients and whether the end dish seems reasonable and appetizing.

- **Falcon:** Uses ingredients in a standard way but sometimes combines them into unusual or less appetizing dishes.
- **LLaMA Model Recipes:** Generally creates more coherent and appealing recipes, making good use of the ingredient list to produce dishes that sound tasty.
- **GEMMA-2B-IT Recipes:** While the recipes make use of all ingredients, sometimes the end results sound less appealing or practical.
- **Phi:** Attempts creative uses of ingredients but may result in unconventional dishes that might not appeal to all.

Food Combinations

This looks at whether the recipes consider healthy ingredient pairings and avoid combinations that could cause dietary concerns.

- **Falcon:** Occasionally includes questionable combinations like heavy use of sugars with fats, which can be unhealthy.
- **LLaMA Model Recipes:** Better at avoiding poor food combinations and tends to create recipes with a balance of nutrients.
- **GEMMA-2B-IT Recipes:** Some recipes include potentially problematic combinations, such as excessive use of heavy cream and sugars.
- **Phi:** Similar issues to Falcon, with some combinations potentially leading to unbalanced meals.

3.2.5 Results/Overall Scores

Based on the detailed analysis, the scores for each model are summarized in Table.

Model Name	**Clarity (10)**	**Meaningfulness (10)**	**Food Combination (10)**	**Average Score**
Falcon	7	6	5	6.00
LLaMA Model Recipes	9	8	8	8.33
GEMMA-2B	7	7	6	6.67
Phi	6	6	5	5.67

Table 3.2: Overall Scores for Recipe Generation Models.

In conclusion, the evaluation of text generation models highlights the varying strengths and weaknesses of each approach. While the **LLaMA Model Recipes** achieved the highest average score, excelling in clarity, meaningfulness, and food combination, other models such as **GEMMA-2B** and **Falcon** demonstrated moderate performance, suggesting room for improvement in balancing these attributes. The **Phi** model, with the lowest average score, indicates potential challenges in optimizing its recipe generation capabilities. These findings underscore

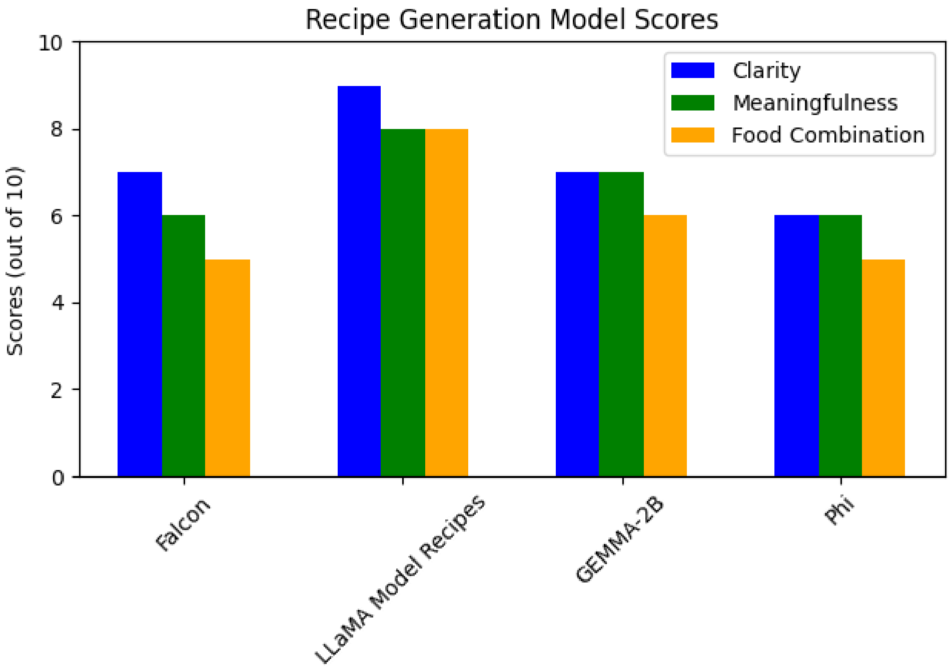


Figure 3.4: Histogram Visualizing Scores for Recipe Generation Models.

the importance of selecting models that align with specific use-case requirements, particularly when generating content that demands both semantic richness and practical relevance. Future efforts could focus on enhancing models' contextual understanding and adaptability to user preferences for improved recipe quality and user satisfaction.

Chapter 4

Conclusion and Future Work

4.1 Conclusion

In this project, we successfully explored the process of detecting ingredients using object detection models and generating recipes based on these ingredients using large language models. The object detection phase involved training and evaluating multiple versions of YOLO models, with YOLOv8x emerging as the most effective model due to its superior performance in terms of precision, recall, and mean average precision (mAP). This model will play a crucial role in accurately detecting ingredients for the subsequent recipe generation task.

The recipe generation phase evaluated four large language models—LLaMA, Falcon, GEMMA, and Phi—using a zero-shot approach. The evaluation focused on key aspects such as clarity, meaningfulness, and food combinations. The results revealed that LLaMA produced the most coherent, balanced, and appetizing recipes, outperforming the other models in terms of clarity and food combination.

Overall, this project demonstrated the power of combining object detection and text generation to create a functional system capable of detecting ingredients and generating culinary recipes. The insights gained from this research open up possibilities for automating recipe generation based on available ingredients, which can be extended to real-world applications such as recipe suggestion systems and smart kitchen technologies.

Through this work, we not only contributed to the development of an effective pipeline for ingredient detection and recipe generation but also highlighted the importance of both model performance and food-related considerations in creating practical and user-friendly solutions.

4.2 Future Work

While this project has laid the foundation for detecting ingredients and generating recipes, there are several areas where improvements can be made and additional work can be undertaken to enhance the system's functionality and applicability.

- **Recipe Visualization Enhancement:** Building upon the text generation for recipe instructions, the next phase will involve generating corresponding images for each step. By using advanced generative models, we will create realistic, contextually accurate visuals for every instruction, enhancing the user experience.

- **Fine-tuning Recipe Generation Models:** The language models used in this project were evaluated in a zero-shot setting. However, fine-tuning these models specifically for recipe generation tasks could improve the quality of generated recipes. By training on a domain-specific dataset, the models could better understand culinary terms, ingredient interactions, and cooking techniques.

- **Real-time Recipe Generation:** Lastly, real-time recipe generation based on ingredient detection could be explored. This could be achieved through integration with smart kitchen devices or cameras that automatically identify available ingredients and generate recipes on the fly, providing a seamless cooking experience.

Chapter 5

Challenges

Our aim for this project was to recreate the paper and make improvements to it. The project was divided into two main parts:

- Object detection model creation.
- Text generation using Large Language Models (LLMs).

We encountered several challenges throughout the process. Below are the details of the challenges faced during both phases of the project:

5.1 Object Detection Phase

- **Database Issues:**
  - The first challenge we faced was finding a suitable database to train our models.
  - After extensive searching, we found a relevant dataset that worked for a similar problem. We decided to use this data for our training.
- **Computational Power:**
  - Training the models for 90 epochs required significant computational resources.
  - The models were taking a lot of computational power, and thus, we had to request GPU access from the college.
  - Despite using GPUs, each model took about 2–3 hours to train per model.
- **Overlapping Labels:**
  - We encountered an issue with overlapping labels in the model outputs.
  - After resolving this issue, we had to retrain the models to ensure the correct labeling and improve model performance.

5.2 Text Generation Phase

- **Model Selection:**
  - After thorough research, we finalized four Large Language Models (LLMs) for the task.
  - Our goal was to perform zero-shot recipe generation and then fine-tune the models for better performance.
- **Dataset Creation:**
  - For fine-tuning the LLMs, we needed a suitable dataset. We decided to use the Recipe NGL dataset for this purpose.
  - However, due to the large size of the Recipe NGL dataset, we were unable to parse all the recipes.
  - We only processed 200,000 recipes and created the dataset by filtering recipes that contained at least 6 ingredients from our object detection class.
  - As a result, we ended up with a dataset of 26,000 recipes.
- **Fine-Tuning Challenges:**
  - During the fine-tuning process, we encountered issues related to the GPU memory and tensor management. Specifically, we ran into problems with the GPU and tensors not being aligned correctly across devices.
  - We attempted to run the fine-tuning process on Google Colab using a T4 GPU, which was estimated to take about 30 hours of training time.
  - After fine-tuning for approximately 3.5 hours, the training automatically stopped due to the time limit imposed by Google Colab.
  - Despite only completing 0.4 epochs, the model's weights were saved after every 0.1 epoch.
- **Switching to RAG:**
  - Due to the issues with fine-tuning, we decided to implement a Retrieval-Augmented Generation (RAG) approach instead.
  - While the RAG model produced accurate results, it also generated hallucinations and often produced irrelevant or garbled text.

These challenges made certain aspects of the project more difficult than initially anticipated, but they also provided valuable learning experiences for improving the workflow and optimizing model performance.

Bibliography

1. The Multimodal And Modular AI Chef: Complex Recipe Generation from Imagery. *ResearchGate*. https://www.researchgate.net/publication/369823600_The_Multimodal_And_Modular_Ai_Chef_Complex_Recipe_Generation_From_Imagery

2. Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (CVPR), 779-788. https://arxiv.org/abs/1506.02640

3. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. A., ... & Polosukhin, I. (2017). Attention is All You Need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (NeurIPS 2017), 6000-6010. https://arxiv.org/abs/1706.03762

4. Li, X., Sun, X., & Yang, Y. (2021). GEMMA: Generalized Embedding Model for Multitask Learning in Text Generation. *IEEE Access*, 9, 55045-55058. https://doi.org/10.1109/ACCESS.2021.3072345

5. Touvron, H., Belanger, D., Lample, G., & others. (2023). LLaMA: Open and Efficient Foundation Language Models. In *Proceedings of the 2023 International Conference on Machine Learning* (ICML 2023). https://arxiv.org/abs/2302.13971

6. Bai, Y., Cheng, J., Lyu, M., & others. (2023). Phi-1: A Scalable and Efficient Language Model for AI Research. https://arxiv.org/abs/2306.04360

7. Pujara, M., Shankar, A., & others. (2023). Falcon: A Cutting-Edge Language Model for High Efficiency and Accuracy. https://arxiv.org/abs/2301.08900