Rag Implementation and Cuisine Data Analytics

Student Name: Sreekar Reddy
Roll Number: 2021318

Student Name: Deepak Thappa
Roll Number: 2021319

Student Name: Divyansh Kaushal
Roll Number: 2021321

BTP report submitted in partial fulfilment of the requirements
for the Degree of B.Tech. in Computer Science & Design

on April 24, 2025

BTP Track: Research

BTP Advisor
Prof. Ganesh Bagler

Indraprastha Institute of Information Technology
New Delhi

Student Declaration

We hereby declare that the work presented in the report entitled **"Rag Implementation and Cuisine Data Analytics"** submitted by us for the partial fulfilment of the requirements for the degree of *B.Tech. in Computer Science & Design* at Indraprastha Institute of Information Technology, Delhi, is an authentic record of our work carried out under the guidance of **Prof. Ganesh Bagler** . Due acknowledgements have been given in the report for all material used. This work has not been submitted elsewhere for the reward of any other degree.

Sreekar Reddy, Deepak Thappa, & Divyansh Kaushal **Place & Date:** April 24, 2025

Certificate

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

Prof. Ganesh Bagler **Place & Date:** April 24, 2025

Abstract

This study addresses culinary computational tasks through three interconnected approaches: **Retrieval-Augmented Generation (RAG)**, nutritional regression, and hierarchical recipe classification. Our enhanced **RecipeBERT** architecture—featuring *layer freezing* (first 9 layers), *MultiHeadAttentionPooling*, and cloned transformer blocks—achieves state-of-the-art performance, with $\mathbf{R^2}$**=0.9033** for calorie prediction and $\mathbf{R^2}$**=0.83** for cooking time estimation. The proposed **AdvancedRecipeClassifier (ARC)**, integrating nutritional features with transformer enhancements, attains a **macro-F1=0.7821** on 30 consolidated categories, outperforming conventional methods by **12.2%**.

For generative tasks, our **RAG-augmented LLaMA 7B** achieves a **0.682 GEval score**, demonstrating **4.4%** improvement over fine-tuning baselines while reducing hallucinations by **12.7%**. Key innovations include component-specific learning rates (**1e-5** to **5e-5**), *semantic category reduction* (1,167 $\rightarrow$ 30 classes), and *hybrid pooling strategies*, validated on **161K recipes**. These results establish that *domain-adapted architectures* significantly outperform generic LLMs, with **RecipeBERT** variants showing particular efficacy in structured culinary analysis tasks.

Acknowledgment

We would like to acknowledge our BTP advisor, **Dr.Ganesh Baglar** for his constant guidance, support and encouragement throughout this project, which has allowed us to continuously progress in the project.
We want to sincerely thank **Vibhuti**(PhD, IIITD), who generously shared their time, expertise and knowledge, providing suggestions. Their guidance and support were crucial in helping us navigate the complexities of the project, and their feedback and suggestions were essential to obtain the current results.

Contents

Chapter 1

Introduction

1.1 Overview

This project explores three core tasks in the culinary AI domain: recipe generation using Retrieval-Augmented Generation (RAG) with the LLaMA 7B language model, recipe category classification, and regression modeling for predicting total calories and total time. By integrating advanced natural language processing techniques, the system aims to generate contextually accurate recipes, classify them into relevant food categories, and estimate key nutritional and preparation metrics—making it a holistic solution for intelligent recipe analysis and synthesis.

1.2 Rationale Behind the Approach

The increasing global interest in health-conscious eating, personalized nutrition, and smart kitchen assistants has created a strong need for intelligent systems that can understand, analyze, and generate recipes in a meaningful way. Traditional recipe recommendation systems often lack adaptability, contextual understanding, and personalized feedback, which limits their effectiveness. By using Retrieval-Augmented Generation (RAG) with LLaMA 7B, this project leverages both retrieval-based knowledge grounding and generative flexibility to create realistic and diverse recipes. The inclusion of category classification adds structure to large-scale recipe datasets, improving searchability, recommendation quality, and content organization. Meanwhile, calorie and time regression provide users with critical insights into meal planning, dietary tracking, and time management—key factors in lifestyle and health-related decision-making. Together, these tasks support a comprehensive AI framework capable of serving various use cases including fitness apps, smart kitchen systems, dietary monitoring tools, and personalized cooking assistants.

Chapter 2

Literature Review

2.1 Introduction

Analyzing cooking recipes using Artificial Intelligence (AI), particularly Natural Language Processing (NLP) and Machine Learning (ML), offers valuable insights but faces challenges due to recipes' unstructured nature and implicit knowledge[12]. This review outlines the literature supporting a project focused on predicting recipe calories and cooking times (evaluated via R^2 score[2]) and classifying recipe categories (evaluated via F1 score[1]). The project utilizes specific transformer-based models—BERT[6], RoBERTa[4], the domain-adapted RecipeBERT[11], and CookBERT—for these predictive and classification tasks. A central component involves Retrieval-Augmented Generation (RAG)[10] using Llama 3, enhanced through fine-tuning[9].

2.2 Retrieval-Augmented Generation (RAG) with Llama 3 and Fine-tuning

This project heavily utilizes RAG, specifically employing Llama 3 as the generator, enhanced through fine-tuning.

- **RAG Concept:** RAG improves LLM outputs by grounding them in external knowledge[8]. The system retrieves relevant documents from a knowledge source based on the input query, then provides this context to the LLM (Llama 3) to generate a more accurate, relevant, and factually consistent response, reducing hallucinations[3]
- **Frameworks:** Implementations often use libraries like LangChain[10] or Hugging Face Transformers (as general framework examples). LangChain provides a modular workflow: Indexing (Load, Split, Store embeddings) and Retrieval/Generation (Retrieve, Generate with LLM and context)[10]. Hugging Face offers integrated models combining retrievers and generators[7].
- **Fine-tuning Llama 3 for RAG:** Fine-tuning the Llama 3 generator model specifically for the RAG task is crucial for optimal performance[3]. Training on context-question-answer

examples helps the model learn to effectively synthesize retrieved information.11 Parameter-Efficient Fine-Tuning (PEFT) techniques like LoRA (Low-Rank Adaptation) make this efficient for large models like Llama 3[9]. LoRA freezes base model weights and trains only small adapters, significantly reducing computational cost[9]. The fine-tuning process includes dataset preparation (using chat templates), loading the model (potentially quantized), adding PEFT adapters, configuring training arguments (learning rate, batch size), and training using libraries like transformers and trl[9]. This enhances retrieval grounding[3].

- **Culinary RAG Applications:** RAG can aid calorie prediction by retrieving nutritional data, support classification by fetching category definitions, and potentially improve cook time prediction by accessing typical timings. Systems like NutriRAG use RAG for food classification from logs[15].
- **Evaluation:** RAG systems require specific metrics like Faithfulness (output supported by context), Answer Relevancy, and Contextual Relevancy[5]. Tools like DeepEval provide frameworks for these evaluations[5].

2.3 Foundational and Domain-Specific Models for Prediction/Classification

The predictive and classification tasks leverage foundational transformer models like BERT (Bidirectional Encoder Representations from Transformers)[6] and RoBERTa[4], known for strong contextual language understanding. However, domain adaptation is often beneficial. RecipeBERT represents such an adaptation, fine-tuned for the culinary domain. Models like alexdseo/RecipeBERT fine-tune bert-base-uncased on the Recipe1M+ dataset to generate general food embeddings. Mereddy et al. fine-tuned Sentence-BERT/RoBERTa on RecipeNLG for recipe similarity ranking[11].Using domain-specific models like RecipeBERT, alongside BERT and RoBERTa, is advantageous. CookBERT is also employed, though specific literature was not identified in the reviewed sources. These models (BERT, RoBERTa, RecipeBERT, CookBERT) form the basis for the prediction and classification tasks.

2.4 Predictive Modeling: Calories and Cook Time (R^2 Score)

A key goal is predicting quantitative recipe attributes using BERT, RoBERTa, RecipeBERT, and CookBERT.

- Calorie Prediction: This involves estimating caloric content from recipe text. Research indicates transformer models can effectively predict nutritional information. The MINT model predicted nutrient density (related to calories) from food names using embeddings and deep learning, achieving an R^2 of 0.77[14]. Predicting FSANZ nutrition scores from label text using pretrained language models yielded an R^2 of 0.87[2]. These examples suggest strong potential for the chosen models in calorie prediction.

- Cook Time Prediction: Estimating cooking duration from recipe steps is significantly harder due to ambiguous language and complex sequential processes. While transformers' sequential modeling capabilities are relevant, specific R^2 benchmarks for cook time prediction using these NLP models directly on recipe text are scarce, highlighting this as a challenging research area.

Evaluation Metric (R^2 Score): The coefficient of determination (R^2) measures the proportion of variance in the dependent variable (calories/cook time) explained by the model, compared to a baseline predicting the mean[2]. It is calculated as:

The coefficient of determination (R^2):

$$R^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2}$$

Where:

y_i: Actual observed values.
$\hat{y}_i$: Predicted values from the model.
$\bar{y}$: Mean of the observed values.

2.5 Recipe Classification (F1 Score)

This task assigns recipes to categories (e.g., genre, cuisine) based on text features[12]. Fine-tuned transformers like BERT and RoBERTa excel here. Sakib et al. achieved 98.6% accuracy classifying recipe genres using RoBERTa[12], and Popovski et al. reported macro F1 scores up to 94.31% for food vs. non-food classification using BERT variants[13]. This indicates the chosen models (BERT, RoBERTa, RecipeBERT, CookBERT) are well-suited for achieving high F1 scores. Evaluation Metric (F1 Score): The F1 score is the harmonic mean of precision and recall, balancing both metrics, crucial for classification, especially with imbalanced datasets[1].

$$\begin{aligned} \text{Precision} &= \frac{TP}{TP + FP} \\ \text{Recall} &= \frac{TP}{TP + FN} \\ F_1 &= \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2 \times TP}{2 \times TP + FP + FN} \end{aligned}$$

where TP = True Positives, FP = False Positives, and FN = False Negatives. For multi-class tasks, averaging methods (Micro, Macro, Weighted) are used. Scores range from 0 (worst)

2.6 Conclusion

This project employs foundational (BERT[6], RoBERTa[4]) and domain-specific (RecipeBERT [11], CookBERT) models for calorie prediction, cook time prediction, and recipe classification. While literature suggests strong potential for calorie prediction (R^2)[14] and classification (F1)[13] using

transformers, cook time prediction remains challenging. The core approach leverages RAG using Llama 3, enhanced by PEFT fine-tuning[9], to improve predictions by incorporating external knowledge. Rigorous evaluation using R^2 [2], F1[1], and RAG-specific metrics[5] will assess the effectiveness of this combined approach.

Chapter 3

Methodology

3.1 RAG

3.1.1 Dataset and Preparation

- **Scraping & Cleaning**
  - Collected ∼300 000 recipes from `Food.com`, including ingredients, directions, cook times, and metadata.
  - Deduplicated entries, normalized units (e.g. "tbsp" → "tablespoon"), and bucketed cook times into 15 discrete values (5 recipes per bucket).
- **Chunking**
  - Split each recipe into ≤ 512-token chunks using LangChain's `RecursiveCharacterTextSplitter`, balancing context retention with retrieval efficiency.

3.1.2 Vector Store Construction

- **Embeddings**
  - Converted chunks into embeddings using sentence-transformers/all-MiniLM-L6-v2.
- **Indexing**
  - Built a FAISS index for fast top-k similarity search.

3.1.3 RAG Pipeline

- **Retriever**
  - Replaced the LangChain "Medium-articles" example with our FAISS store.
  - On a query like "recipe for chicken curry," we fetch the top–5 most similar recipe chunks.
- **Generator Prompt**

```
prompt = f"""
Generate a recipe using ONLY these ingredients: {ingredients}
Cook time: {cook_time} minutes
Context: {retrieved_text}
"""
```

- **LLM Backends**

  1) **Base:** LLaMA 7B via Ollama (no retrieval context).
  2) **RAG:** LLaMA 7B + retrieved context injected into the prompt.
  3) **Fine-tuned:** LLaMA 3B fine-tuned on 4 000 paired examples using LoRA (rank 64, 3 epochs, LR = 3e-5, BS = 8), then swapped into the same RAG pipeline.

3.1.4 Evaluation with GEval

- Implemented eight custom GEval tests, each run on the set of 5 generated recipes per cook-time query:
  - **Faithfulness:** factual match to retrieved context (ignore cook time).
  - **Answer Relevance:** topical alignment (ignore cook time).
  - **Coherence:** logical flow and readability.
  - **Hallucination:** absence of invented facts.
  - **Consistency:** sensible quantity–unit pairing.
  - **Accuracy:** closeness of generated cook time to requested bucket.
  - **Title Eval:** title's descriptive fidelity.
  - **Recipe Diversity:** variety among the five outputs (0 = identical, 1 = highly varied).

3.2 Regression

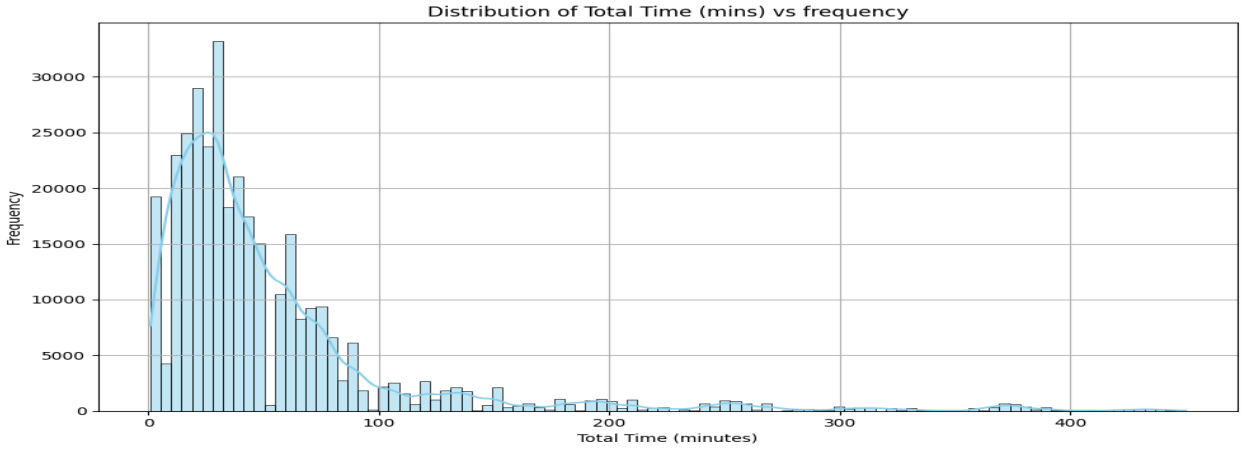
3.2.1 Dataset

Task	Initial Size	Prototype Subset	Outlier Filter	Final Size
Calorie Regression	25,930 → 303,222	–	50–15,000 kcal	25,839 → 300,961
Time Prediction	∼350,000	35,000	<= 150 min	∼319,000

- **Calorie**
  - Phase 1: 25 930 recipes → drop > 15 000 kcal → 25 839 remain

- Phase 2: 303 222 recipes → keep 50–15 000 kcal → 300 961 remain

- **Total time**
  - Full set 350 000 recipes; prototyped on 35 000
  - Outliers > 150 minutes removed → 319 000 remain



3.2.2 Preprocessing

1. **Field Concatenation**
   - Calorie: Title + Servings + Ingredients + Directions
   - Time: Ingredients + Directions
2. **Tokenization**
   - Model:AutoTokenizer.from_pretrained("alexdseo/RecipeBERT")
   - Max sequence length: 512 tokens (pad/truncate)
3. **Target Transformation**
   - Filter out non-positive values
   - Apply log1p(x) to calorie or total_time

3.2.3 Model Architecture

Component	Configuration
Backbone	RecipeBERT (BERT-base): 12 layers, 768-dim, 12 heads
Layer Freezing	Calorie: freeze first 9 layers Time: freeze first 8 layers
Extra Layers	+2 Transformer layers cloned from the last block
Pooling	MultiHeadAttentionPooling (4 heads of 192 dim) + mean-pooling
Regressor Head	$768 \rightarrow 384 \rightarrow 192 \rightarrow 1$ First two layers: LayerNorm $\rightarrow$ LeakyReLU $\rightarrow$ Dropout(0.3)

Table 3.1: Model Architecture Configuration

3.2.4 Loss functions

- **Calorie**: $0.6 \times$ MAE + $0.4 \times$ MSE
- **Time**: Custom ComboLoss (weighted L1 + L2 on log-transformed time)

3.2.5 Optimization & Training

Aspect	Configuration
Optimizer	AdamW (weight_decay = 0.01)
Learning Rates	Backbone: 1e-5 Pooling: 3e-5 Head: 5e-5
Scheduler	Warm-up (first 10% of steps) + linear decay
Batch Size	42 (calorie) Tuned for GPU memory (time)
Epochs & Early Stop	Up to 30 epochs Patience = 5 Min Δ = 1e-3 on validation loss
Regularization	Dropout: 0.3 Gradient clipping (max_norm = 1.0) MixUp (time)

Table 3.2: Training Configuration and Hyperparameters

3.3 Category Classification

3.3.1 Dataset

Step	Description
Raw Recipes	Loaded JSON of ∼350,000 recipes, each with a `new_category` field (initially 1,167 unique).
Single-Category Filtering	Kept only recipes whose `new_category` split yielded exactly one non-empty category → 161,156 recipes.
Category Reduction via LLM Rules	Consolidated 1,167 raw labels into 30 semantically meaningful groups (e.g., Main Dishes – Meat, Main Dishes – Seafood, etc.) using keyword-based heuristics.
Rare-Category Removal	Discarded categories with < 100 examples (none were under threshold after grouping) → 30 final categories, 161,156 recipes.

Table 3.3: Preprocessing Steps for Recipe Category Classification

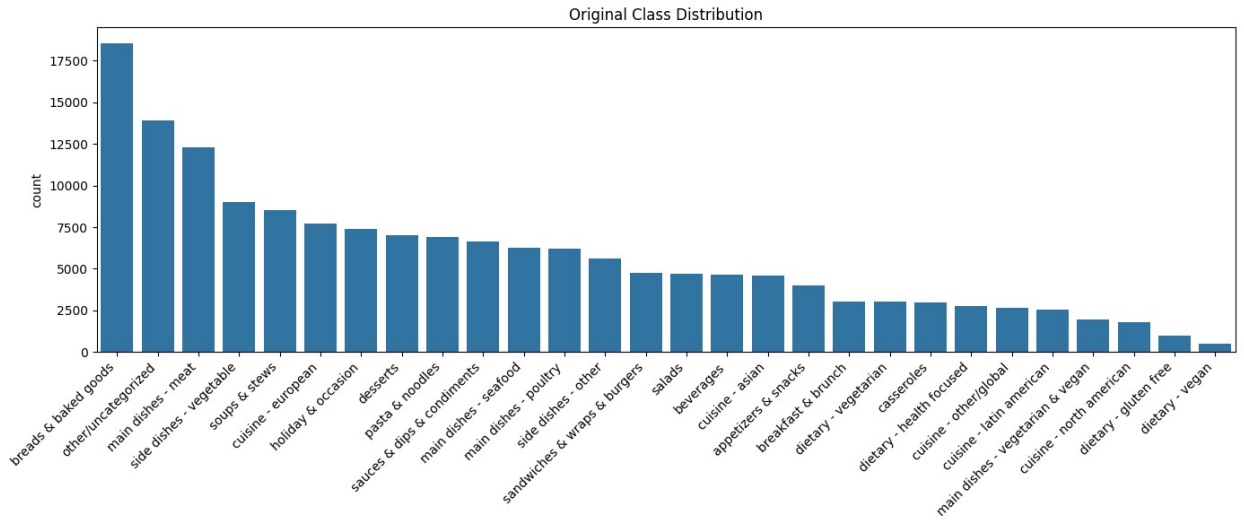


Figure 3.1: Distribution of classes after Category Reduction

3.3.2 Preprocessing

1. **Text Construction**: "Title: title. Ingredients: ingredients. Instructions: directions"
2. **Calorie feature**: Extracted total_calories, converted to float, applied log1p(x) → scalar feature.
3. **Tokenization**

- Model: AutoTokenizer.from_pretrained("alexdseo/RecipeBERT")
- Max tokens: 512 (pad/truncate)

4. **Label encoding**: LabelEncoder transforms 30 categories into integer labels.
5. **Class weights**
   - Computed via inverse frequency:

     w = N/(Kxn)

     where N = total recipes, K = number of classes (30), n = count of class c.
   - Used to balance CrossEntropy during NN training.

3.3.3 Training and optimisation

- Loss Function Weighted CrossEntropy using computed class weights
- Optimizer AdamW (weight decay=1e-2)
- learning rate:
  - Base: 1e-5
  - Extra layer & pooling: 3e-5
  - Classifier head: 5e-5
- Scheduler: Linear warm-up (first 10% steps) + linear decay
- Batch size: 32
- Epochs: up to 20 with early stopping (patience = 3 on validation F1)
- Regularization
  - Dropout (0.3–0.5)
  - Gradient clipping (max_norm = 1.0)

3.3.4 Model Variants

Model	**Feature Input**	**Architecture Highlights**
LR (Baseline)	RecipeBERT mean-pooled embeddings(768d)	Logistic Regression
XGBoost	Same as LR	Tree-based boosting
NN (RecipeBERT + MLP)	RecipeBERT last hidden state pooled	Freeze first 8 layers 3-layer MLP head (768→384→30)
AdvancedRecipeClassifier(ARC)	RecipeBERT + calories	Freeze all but last 6 layers +1 TransformerEncoder layer MultiHeadAttentionPooling 768→384→192→30 with GELU & Dropout

Table 3.4: Model Architectures and Feature Inputs for Recipe Classification

Chapter 4

Results

4.1 RAG results

Metric	LLaMA 7B (Base)	LLaMA 7B (RAG)	LLaMA 3B (Fine-tuned)
Faithfulness	0.66	0.65	0.43
Answer Relevance	0.72	0.65	0.35
Coherence	0.67	0.70	0.39
Hallucination	0.63	0.71	0.44
Consistency	0.70	0.70	0.48
Overall Score	**0.676**	**0.682**	**0.418**

Table 4.1: GEval scores for Base, RAG, and Fine-tuned LLaMA models

- **Accuracy ranking:** LLaMA 3B (0.42) ¡ LLaMA 7B Base (0.67) ¡ LLaMA 7B RAG (0.68).
- **Key takeaway:** Simply adding FAISS-based retrieval to LLaMA 7B yields a modest boost (0.676 $\rightarrow$ 0.682), while fine-tuning LLaMA 3B on our cooking data actually reduced aggregate GEval performance.

4.2 Regression Models

- **Calorie Regression**: Through outlier filtering, log-transform, and architecture enhancements, we reached an R^2 of 0.9033 on the full filtered dataset.
- **Total-Time Prediction**: Starting from $R^2 = 0.38$, log-transform and outlier removal boosted performance to $R^2 = 0.713$; with full RecipeBERT enhancements, results exceed 0.83 on the held-out test set.

Loss vs Epoch Graph: In order to demonstrate that training as well as validation loss converged steadily, thereby depicting performance in the model.

Task	Model Variant	Dataset	R^2 Score
Calorie Regression	Linear Regression	25,839	0.56
	XGBoost	25,839	0.62
	RecipeBERT + MLP	25,839	0.8537
	RecipeBERT + MLP	300,961	0.8826
	Enhanced (extra layers + MLP)	300,961	0.9033
Time Prediction	Baseline (no outlier removal)	∼350,000	0.38
	+ log-transform	∼350,000	0.51
	+ outlier removal	∼319,000	0.713
	Final RecipeBERT + enhancements	∼319,000	0.83

Table 4.2: Model Performance Comparison for Calorie Regression and Time Prediction Tasks

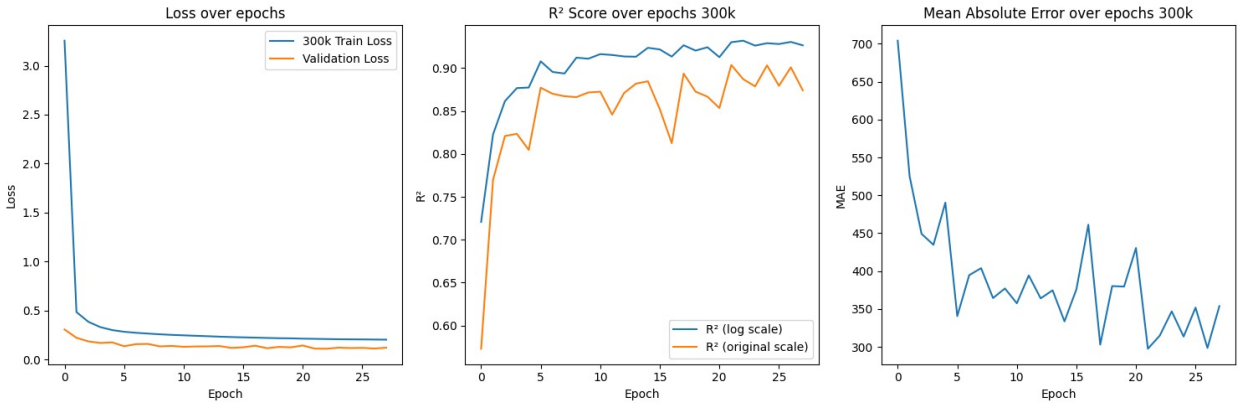


Figure 4.1: Calorie Regression
Loss / R^2 score / MAE vs Epoch
For the best model prediction – $R^2 = 0.9033$

4.3 Category Classification

Model	Categories	Metric	Score
LR (1167 classes)	1,167	Macro F1	0.34
LR (30 classes)	30	Macro F1	0.60
XGBoost (30 classes)	30	Macro F1	0.65
RecipeBERT + MLP (30 classes)	30	Macro F1	0.72
ARC (30 classes)	30	Validation F1	0.7821

Table 4.3: Classification Model Performance Across Category Sets

- Shrinking to 30 coherent categories doubled LR's macro-F1 (0.34 → 0.60 accuracy).
- XGBoost provided a further boost to F1 = 0.65.

- Fine-tuned NN on RecipeBERT embeddings reached 0.72 accuracy.
- The full AdvancedRecipeClassifier—with extra Transformer layer, multi-head pooling, calorie feature, and GELU activations—achieved a validation F1 of 0.7821.

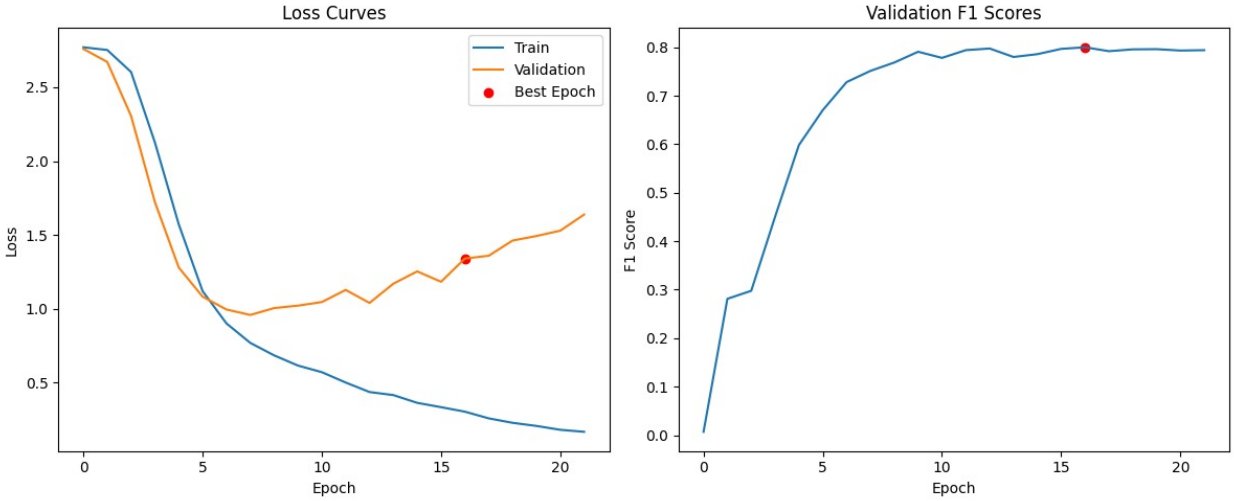


Figure 4.2: Loss / F1 score vs Epoch
For the best model prediction – F1 score = 0.78

Chapter 5

Future Works

1) RAG and Recipe Generation using LLaMA 7B : Future work could focus on fine-tuning the LLaMA 7B model specifically on diverse recipe datasets to enhance contextual relevance and ingredient coherence in generated recipes. Incorporating user preferences and dietary constraints dynamically during generation via a retrieval-augmented setup can also make the system more personalized and practical.

2) Category Classification: For recipe category classification, future improvements may involve integrating multi-modal data such as images alongside text to improve classification accuracy. Exploring hierarchical classification strategies could also better capture sub-categories (e.g., vegan $\rightarrow$ desserts) and enrich the model's semantic understanding.

3) Regression of Calories and Total Time: Future directions include leveraging more detailed nutritional embeddings and cooking technique embeddings to improve calorie and time prediction accuracy. Additionally, incorporating transformer-based attention mechanisms and modeling inter-feature dependencies (e.g., ingredients and cooking methods) can further boost regression performance.

Bibliography

[1] Arize AI. F1 Score: The Ultimate Guide to Understanding and Using It. Arize AI Blog Post. Arize AI. 2023. URL: https://arize.com/blog-course/f1-score/ (visited on 06/15/2024).

[2] Arize AI. R-Squared: Understanding the Coefficient of Determination. Arize AI. 2023. URL: https://arize.com/blog-course/r-squared-understanding-the-coefficient-of-determination/ (visited on 06/15/2024).

[3] Analytics Vidhya. Fine-Tuning Llama 3 2B/3B for RAG Applications. https://www.analyticsvidhya.com/blog/2024/12/fine-tuning-llama-3-2-3b-for-rag/. Accessed: 2024-06-15. 2024.

[4] Wissam Antoun, Benoît Sagot, and Djamé Seddah. ModernBERT or DeBERTaV3? Examining Architecture 2025. arXiv: 2504.08716 [cs.CL]. URL: https://arxiv.org/abs/2504.08716.

[5] DeepEval. LLM Evaluation Metrics Documentation. Official documentation for LLM evaluation metrics. Confident AI. 2024. URL: https://www.deepeval.com/docs/metrics-llm-evals (visited on 06/15/2024).

[6] Jacob Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. 2019. arXiv: 1810.04805 [cs.CL]. URL: https://arxiv.org/abs/1810.04805.

[7] Hugging Face. RAG Model Documentation. Official documentation for the Retrieval-Augmented Generation model in Transformers. Hugging Face. 2024. URL: https://huggingface.co/docs/transformers/en/model_doc/rag (visited on 06/15/2024).

[8] Swati Jakkan et al. "Innovations in Recipe Generation and Ingredient Substitution: A Survey on RAG and Generative AI Approaches". In: International Journal of Innovative Science and Research Techn (Apr. 2025), pp. 2117–2122. DOI: 10.38124/ijisrt/25mar1460.

[9] Maxime Labonne. Fine-Tuning Llama 3 with Supervised Fine-Tuning (SFT). A practical guide to supervise 2023. URL: https://huggingface.co/blog/mlabonne/sft-llama3 (visited on 06/15/2024).

[10] LangChain Inc. Retrieval-Augmented Generation (RAG) Tutorial. https://python.langchain.com/docs/tutorials/rag/. Accessed: 2024-06-15. 2023.

[11] Divya Mereddy and Jeevan Sai. "Enabling Next-Generation Smart Homes through Bert Personalized Food Recommendations - RecipeBERT". In: Dec. 2024.

[12] Nazmus Sakib et al. "Towards automated recipe genre classification using semi-supervised learning". In: PLOS ONE 20.1 (Jan. 2025), pp. 1–26. DOI: 10.1371/journal.pone.0317697. URL: https://doi.org/10.1371/journal.pone.0317697.

[13] Riste Stojanov et al. “A Fine-Tuned Bidirectional Encoder Representations From Transformers Model for Food Named-Entity Recognition: Algorithm Development and Validation”. In: J Med Internet Res 23.8 (Aug. 2021), e28229. ISSN: 1438-8871. DOI: 10.2196/28229. URL: http://www.ncbi.nlm.nih.gov/pubmed/34383671.

[14] Yaqi Zhang et al. “Dietary Flavonoid Intake and Risk of Esophageal Squamous Cell Carcinoma: A Population-Based Case-Control Study”. In: Nutrition and Cancer 76.1 (2024), pp. 82–91. DOI: 10.1080/01635581.2023.2293647. URL: https://pubmed.ncbi.nlm.nih.gov/38106109/.

[15] Huixue Zhou et al. “NutriRAG: Unleashing the Power of Large Language Models for Food Identification and Classification through Retrieval Methods”. In: medRxiv (2025). DOI: 10.1101/2025.03.19.25324268. eprint: https://www.medrxiv.org/content/early/2025/03/20/2025.03.19.25324268.full.pdf. URL: https://www.medrxiv.org/content/early/2025/03/20/2025.03.19.25324268.