The Metropolitan University of Hong Kong
School of Science and Technology

BSc (Hons) in Data Science and Artificial Intelligence
STAT S461F Data Science Project

2024-2025

Object Detection of Food Nutrition Label

By

Kwok Wai Chun,

Lu Yuk Tong,

Ng Sing Man,

Chu Sik Hin,

Ng Chiu Cheuk,

Chow Chun Ting

Supervisor: Mr. Jimmy Kang

Date: 24 April 2025

Acknowledgements

Thank Mr. Jimmy Kang for his guidance

Content

1. Introduction

1.1 Project Background and Objectives

Due to health consciousness rising, with 40% of global consumers prioritizing nutrition in food choices (Nielsen, 2024). The aim of this project is to address the problem of supporting individuals who have diet sensitivity and health-related issues by developing a combination of technological integration with a health awareness approach. The main objectives include focusing on solving the challenges caused by integrating object detection systems into recommendation systems to provide personal food guidance, formulating an enhanced formula and algorithm capable of recommending the best choice of diet for its end-users, and thereby facilitating users in maintaining controlled diets and a healthy physique.

1.2 Project Significance

At 7-Eleven stores in Hong Kong, food recognition software identifies items, helping consumers choose healthier options. This technology utilizes Artificial Intelligence, which employs sophisticated algorithms that can detect and locate objects in images or videos. One instantiation of this technology is the YOLO v4 tiny algorithm. Recommended systems are driven by data science. Hybrid methods use collaborative filtering (which considers user behavior) with content-based filtering (which considers item attributes). These tend to yield accurate suggestions. Knowledge-based systems use explicit health-based rules to provide recommendations. These technologies help create personalized dietary metrics that can recommend healthy, affordable foods and fight the myth of ‘healthy is more expensive’ while promoting intentional eating for food impaired individuals.

1.3 Project Execution

This project can be realized by developing an object-detection system capable of detecting food items accurately and determining their nutritional contents; devising a recommendation algorithm capable of analyzing individual dietary needs and preferences with a view to suggesting a list of the most appropriate food alternatives; and integrating object-detection and recommendation systems so that the user experience in the application is smooth. The products of 7-Eleven outlets (7-Select) aimed at guaranteeing the accessibility of suggested healthier food options, thereby facilitating ease of access for the intended demographic. To enhance the system's efficacy and confirm its alignment with the requirements of individuals experiencing dietary sensitivities and health-related issues, predict user action by his order will be implemented. This might help solve the problem through integration, bringing the best options for the users and finally helping in dietary management about health and wellness concerns for its target population in Hong Kong.

2 Literature Review

2.1 Object detection: Overview and applications

Object detection aims to find the location and category of all interesting objects in images or 3D data; often, the results are in the form of bounding boxes (Gui et al., 2024). In this sense, it is essential to detect food nutrition labels and solve the problem because they are typically small and in different orientations.

2.1.1 One-Stage vs. two-stage detectors: speed and accuracy comparison

Object detection models come in two categories: one-stage and two-stage, which have unique characteristics such as speed and accuracy.

2.1.1.1 Two stage Detectors

Two-Stage Detectors, for example, a detection named Faster R-CNN (Ren et al., 2015) proceeds in two stages, with the first consisting of proposal generation, for instance, by means of algorithms such as selective search or proposal network which reduce the search space to second where boosts are refined to obtain final bounding boxes and class labels. They achieve high accuracy as the refinement step improves localization and classification, while such a procedure is very expensive since it more than doubles the computation time, and thus usually cannot be used in real-time.

2.1.1.2 One stage Detectors

One-Stage Detectors. Models like YOLO (You Only Look Once), These models can directly predict the likelihood of the classes and position of the bounding boxes of objects in the input image during a single pass without any region proposals, making them significantly faster. (Redmon et al. 2016.) From a paper by Lanmei Wang et al.(2021) indicated that YOLOv4-Tiny is “a lightweight version of the YOLOv4 model” that “reaches about 87 frames per second”. However, one-stage detectors might still have slightly lower accuracy for small or closely packed objects because they miss out on an improvement step.

2.1.2 Efficiency Evaluation for Real-Time Applications

Efficiency in object detection refers to the ability to process images quickly with acceptable accuracy in resource-constrained environments, such as mobile or embedded devices. For real-time applications like surveillance or nutrition label detection, a high Frames Per Second (FPS) rate is critical to ensure a seamless user experience. According to Jiang et al. (2020), YOLOv4-tiny achieves a remarkable 371 FPS on a 1080Ti GPU, making it suitable for real-time applications. Its optimizations include the use of Feature Pyramid Networks (FPN) for multi-scale detection for improved bounding box selection, which enhance detection speed without significantly compromising accuracy.

In the context of our project, efficiency is the top priority as users expect rapid responses when scanning nutrition labels. The system may need to operate on devices with limited computational power, such as smartphones. Optimization techniques like model compression and efficient feature scale management are essential to achieve real-time performance.

2.1.3 Impact and Considerations for Selecting Object Detection Methods

Selecting an object detection method for nutrition label detection involves several considerations:

Accuracy	The model must accurately detect nutrition labels despite variations in size, font, and clarity. Misidentification or missed detections could lead to incorrect nutritional data, undermining the system’s reliability.

Real-Time Performance	This processing ensures usability, supports robustness in dynamic environments, which is critical for safety and efficiency in practical deployments.
Small Object Detection	Nutrition labels often occupy a small portion of an image, requiring models optimized for small object detection.
Robustness	The model must perform consistently across diverse conditions, including varying lighting, packaging materials, and label orientations.

By prioritizing lightweight, efficient models like the improved YOLOv4-tiny, and incorporating optimization techniques, the system can achieve real-time, accurate, and robust nutrition label detection.

2.2 Recommender Systems: Overview and Applications in Practice

Recommender systems are advanced algorithms that provide personalized suggestions for items like products, media, or educational resources, addressing information overload in digital environments. These systems are widely used in e-commerce, entertainment, education, and social media to enhance user engagement. Ricci et al. (2021) suggest that core approaches include Content-Based Filtering (CBF), Collaborative Filtering (CF), and Hybrid Filtering (HBF), each applied across domains like movies, music, and online learning.

2.2.1 Content-Based Filtering

Content-Based Filtering recommend products by matching their features to past tastes of a consumer. Javed et al. (2021) suggests that the main benefits of CBF are transparency, clear justifications for recommendations derived from item properties, enabling suggestions for new or unrated items without requiring user interaction data. Systems including news and e-learning benefit from CBF. In the other side, CBF may limits performance when item descriptions are inadequate or overspecialization, recommendations may show a lack of diversity. These problems can make the system less able to identify complex user interests.

2.2.2 Collaborative Filtering

Collaborative Filtering generates recommendations by analyzing user-item interactions, leveraging patterns in user preferences to predict interests. Papadakis et al. (2022) suggest that CF's advantages include its "agnosticity (no domain specific knowledge required for its use". CF has become flexible for use in different areas including e-commerce and streaming platforms. However, CF has certain disadvantages, including data sparsity, most users evaluate just a small selection of objects, which results in sparse matrices that reduce recommendation quality, and the cold start problem, new users or products lack of adequate data for correct suggestions.

2.2.3 Association Rule Analysis

Association Rules (AR) in recommender systems are used to identify patterns in user-item interactions by leveraging transaction data to uncover relationships between items, then enabling personalized recommendations. AR is a technique that analyzes user behavior, including purchase histories, to identify frequent item sets—combinations of items that are frequently selected together as described by Song et al. (2021). These item sets are employed to produce rules (e.g., the probability that a user will select item Y if they select item X) via confidence (reliability of the rule) and support (frequency of item co-occurrence).

2.2.4 Recommendation Based on Similarities or Distance Metrics

Hasan and Ferdous (2024) in their research published in the Journal of Computer Science and Technology Studies suggest a hybrid movie recommendation system that incorporates the Alternating Least Squares algorithm and cosine similarity. Hasan and Ferdous Methodology employ cosine similarity to calculate the cosine of the angle between feature vectors. Cosine similarity facilitates the accurate identification of related products or users within extensive datasets and mitigates challenges such as data sparsity and cold-start problems when combined with hybrid filtering techniques.

2.2.5 Hybrid Methods in Recommendation

Hybrid recommendation methods have become popular for their capacity to mitigate the drawbacks of individual approaches through the integration of many methodologies. AL Fararni et al. (2021) introduced a hybrid tourist recommendation system that combines content-based, collaborative, and context-aware filtering to improve personalization and address data sparsity. Their architecture utilizes big data and artificial intelligence to create dynamic travel itineraries, tackling issues like cold start and overspecialization. Li et al. (2021) designed a hybrid approach that integrates content-based and collaborative filtering, employing K-means clustering by extracting project features, establishing user interest model and calculating the similarity between users. Their experiments enhanced efficacy compared to traditional techniques, especially in managing sparse datasets. These studies underscore the efficacy of hybrid methodologies in achieving a balance among accuracy, scalability, and adaptability, making them particularly suitable for various recommendation contexts.

2.3 Case Study: Landario

Landario , a German eyewear company, uses AI-powered facial recognition to recommend glasses based on users' skin tone and face shape from uploaded selfies in controlled conditions. User-uploaded selfies are used to determine frame colors that match each person's face features. After investigation, Landario's program recommends glasses and explains why these products fit the user. This technique shows how object detection and recommender systems may transform in e-commerce customization, which can be a reference of our research applies to food nutrition label analysis.

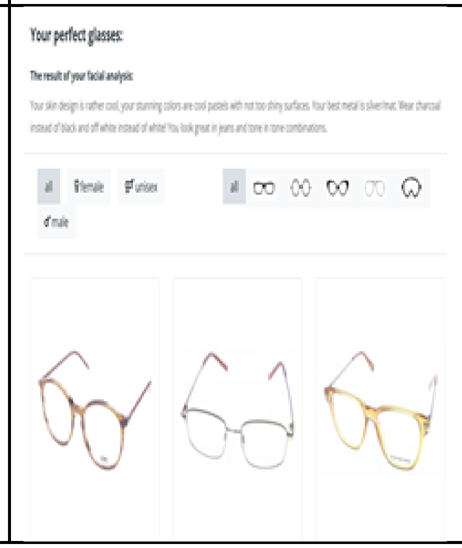
Landario StyleIQ	Submitting Photo	Recommendation
		

Table1: Landario StyleIQ Demonstration

2.3.1 Relationship to This Project

While Landario focuses on glasses recommendations from facial features detection, our project applies a similar pipeline to nutritional recommendations:

Detection: To provide individualized eyewear recommendations, Landario uses AI-powered facial recognition to examine important facial characteristics like skin tone and face shape. Similar to this, our project makes use of YOLOv4-Tiny, a lightweight object detector that has been optimized to recognize and extract nutritional data from food labels. This ensures high accuracy and efficient real-time processing for dietary recommendation systems.

Recommendation: By matching facial features such as cool skin tones with frame colors and styles from their catalog. By contrast, our project uses cosine similarity to align detected food labels with user dietary profiles and association rule to find frequent nutrient pairings. This allows for personalized nutrition recommendations, such as low-sodium options for users with hypertension.

Landario prioritizes visual balance when evaluating recommendations based on subjective user feedback regarding frame suitability derived from facial analysis. In contrast, our project uses quantitative metrics to measure recommendation accuracy and dietary relevance for personalized food suggestions. These metrics include A/B testing and nutrition scoring based on detected label data (e.g., calories, allergens) and user health profiles.

Our case study for a model of excellent integration between object detection and recommendation systems will be Landario. Analyzing Landario's approach on privacy protection, detection accuracy, and suggestion modification helps us to add design components into our model. This case study approach helps us to fit our field of knowledge by allowing us to modify a tested framework.

3. System Overview

3.1 Dataset (7-SELECT)

The synthetic dataset is generated from raw data of 7-SELECT products, tailored for training the object detection and recommendation system. It comprises 20,000 images of 15 unique 7-SELECT products, covering 120 combinations of snacks, beverages, and ready-to-eat meals. Each image is annotated with bounding boxes and labels, capturing product names, nutritional information (e.g., energy, protein, fats, carbohydrates, sugars, sodium), and metadata. To enhance model robustness, the dataset incorporates variations in lighting, angles, and backgrounds. Image synthesis techniques, including 3D modeling and semantic cropping will be discussed in greater technical detail in the method design section. The dataset's diversity and preprocessing ensure accurate detection and reliable nutritional data extraction under varied conditions.

3.2 Detection (YOLOv4-Tiny)

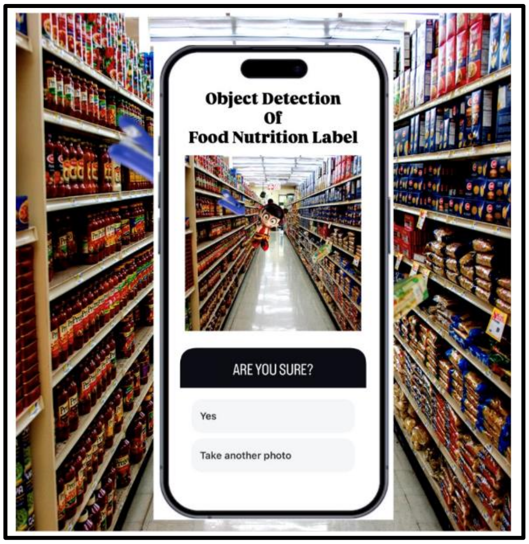
The system utilizes YOLOv4-Tiny, a lightweight one-stage object detection model, to identify 7-SELECT products and extract nutritional data from scanned images. Trained on the 7-SELECT dataset, YOLOv4-Tiny combines features from multiple scales to detect small objects efficiently, achieving high accuracy with low computational requirements. The model processes input images to classify products and retrieve corresponding nutritional information from a pre-stored database, mapping detected labels to metadata. Despite challenges with similar-looking items causing detection confusion, the model's performance is optimized for real-time applications, balancing speed and precision. Metrics such as accuracy, precision, recall, and F1-score guide its evaluation, ensuring reliable integration with the recommender system. More information will be given in the method design part. Further technical details and implementation specifics will be provided in the method design section.

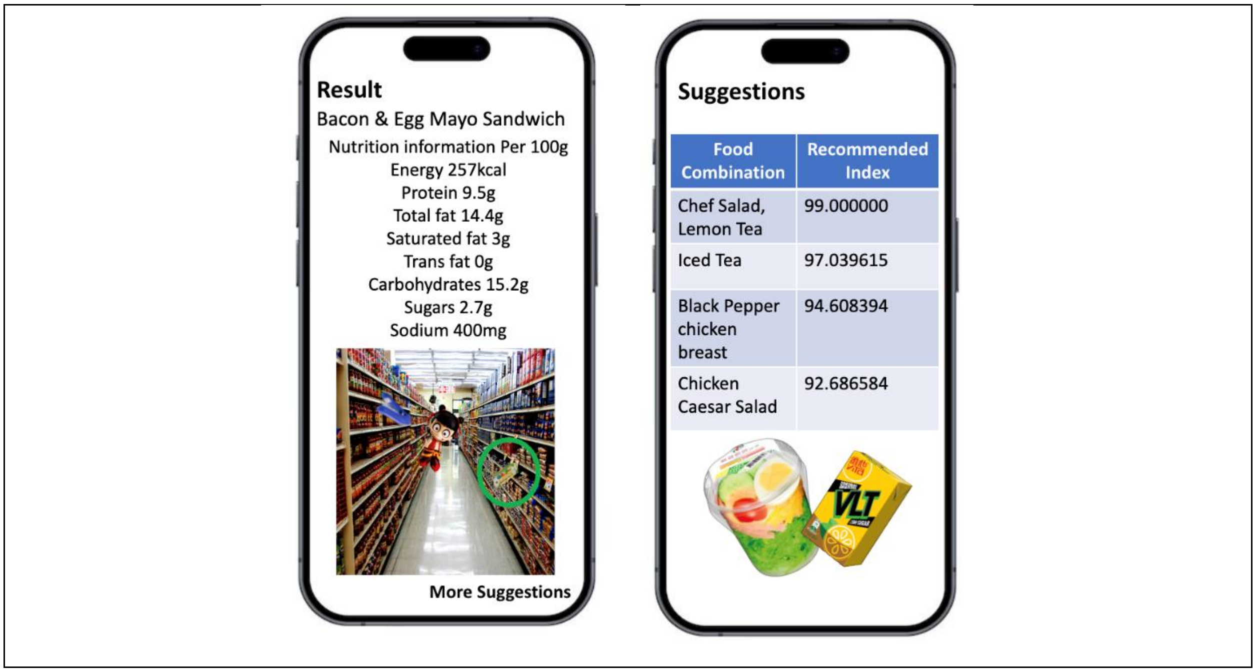
3.3 Recommender System

The recommender system uses a hybrid algorithm to suggest personalized 7-SELECT food options. It combines content-based filtering, leveraging nutritional profiles, with rule-based methods to align suggestions with user dietary goals, such as low sugar or high protein. Products are ranked based on nutritional weights to promote healthier choices. The specific control mechanisms and rule-based methods will be detailed in the method design section.

3.4 Demonstration

This section presents an overview of the system demonstration, showcasing the practical application of the object detection and recommendation system in a real-world scenario. The demonstration was conducted in a supermarket environment to simulate typical user conditions, where background noise, such as shelves stocked with various items and small objects, introduces challenges for accurate detection. The primary objective was to evaluate the system's ability to identify a target product and provide relevant nutritional information and personalized recommendations.





For this demonstration, a photo was captured focusing on a "Bacon & Egg Mayo Sandwich" placed among other items to replicate a cluttered setting. Upon confirming the image through the app interface, the system processed the photo and successfully identified the target product. The app then displayed the nutritional details of the sandwich, providing key information such as energy content, protein, fats, and sodium levels. Following this, the recommendation feature was activated, generating a list of complementary food pairings based on the identified product. The top recommendation, a combination of Chef Salad and Lemon Tea, was highlighted for its high compatibility score, demonstrating the system's capability to offer practical and health-conscious suggestions. This demonstration underscores the system's effectiveness in integrating object detection with personalized recommendations, addressing real-world challenges in food selection.

4. Method & design

4.1 Methodology

4.1.1 Object Detection

Our object detection component is powered by the YOLOv4-tiny model, a compact and efficient version of the YOLO (You Only Look Once) real-time object detection framework. We chose YOLOv4-tiny for its optimal trade-off between detection accuracy and processing speed, making it ideal for applications requiring fast analysis of visual data, such as identifying objects in real-time scenarios. The model employs a simplified architecture with convolutional layers for feature extraction, route layers for multi-scale feature integration, and dual YOLO layers to predict bounding boxes and class probabilities across different object sizes. This setup ensures effective detection of diverse objects in input images. For an in-depth explanation of the model's design and output processes, please see the method design section, specifically Figures 1 and 2.

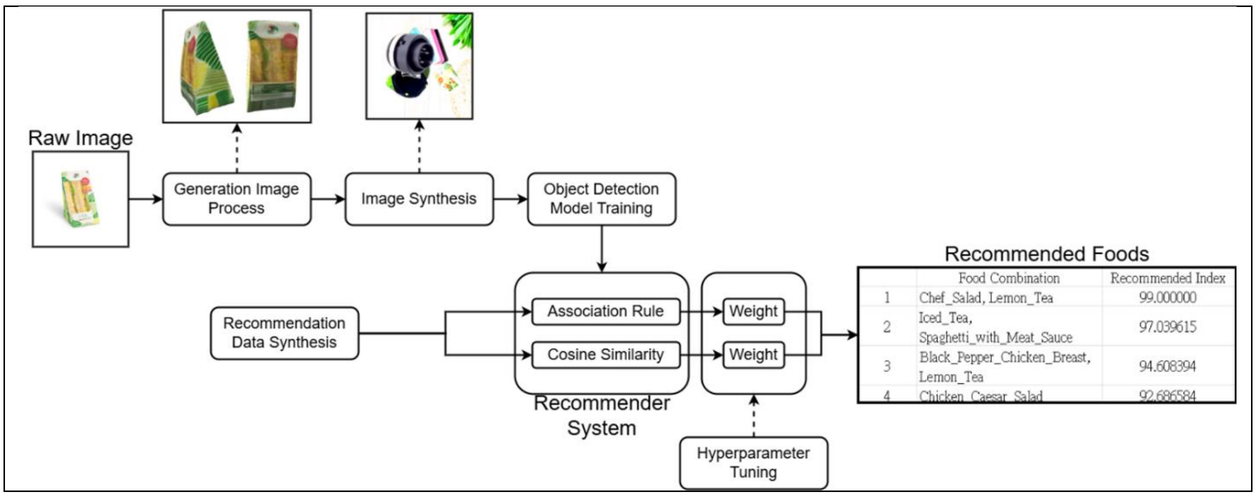
4.1.2 Recommender System

Our recommender system is designed to provide personalized food combinations that meet users' nutritional needs while minimizing reliance on explicit user information. The methodology consists of two main components cosine similarity and association rules. In the Cosine Similarity component, we efficiently identify food combinations based on the nutrients requiring supplementation by utilizing a KD-tree structure. This approach significantly reduces the computational burden associated with exhaustive similarity calculations. We then compute similarity scores to assess the proximity of food combinations to the target nutrients. In the Association Rules component, we employ the FP-Growth algorithm to mine patterns from synthetic transaction data. To generate this synthetic data, we establish rules that simulate eating habits.

Finally, we conduct hyperparameter tuning by randomly assigning weights to both cosine similarities and association rules. The final recommendation score is derived from a weighted combination of these metrics. This structured approach ensures that our system delivers relevant food combinations that effectively address nutritional gaps while minimizing unsuitable suggestions.

4.2 Method design

4.2.1 Overall Architecture



The architecture of our integrated system for food detection and recommendation comprises a suite of interconnected modules: synthetic data generation, object detection, and a recommender system. The operational pipeline commences with the acquisition of raw images extracted from the official 7-Eleven website, predominantly featuring front-facing representations of food items. To address the inherent limitations posed by this singular perspective, Sudo AI were employed to synthesize multi-angular visual data, thereby augmenting the dataset with a broader spectrum of representational diversity.

This enriched synthetic dataset was subsequently integrated with the original corpus, enhancing the robustness and generalizability of the object detection framework. The identified food items are then channeled into the recommender system, which employs association rule mining and cosine similarity metrics to perform a detailed analysis of the detected entities. A sophisticated weighting algorithm was implemented to compute the nutritional profiles of the identified foods, culminating in the generation of a prioritized list of recommended food combinations, optimized for nutritional equilibrium and compatibility with the detected items.

4.2.2 Object Detection

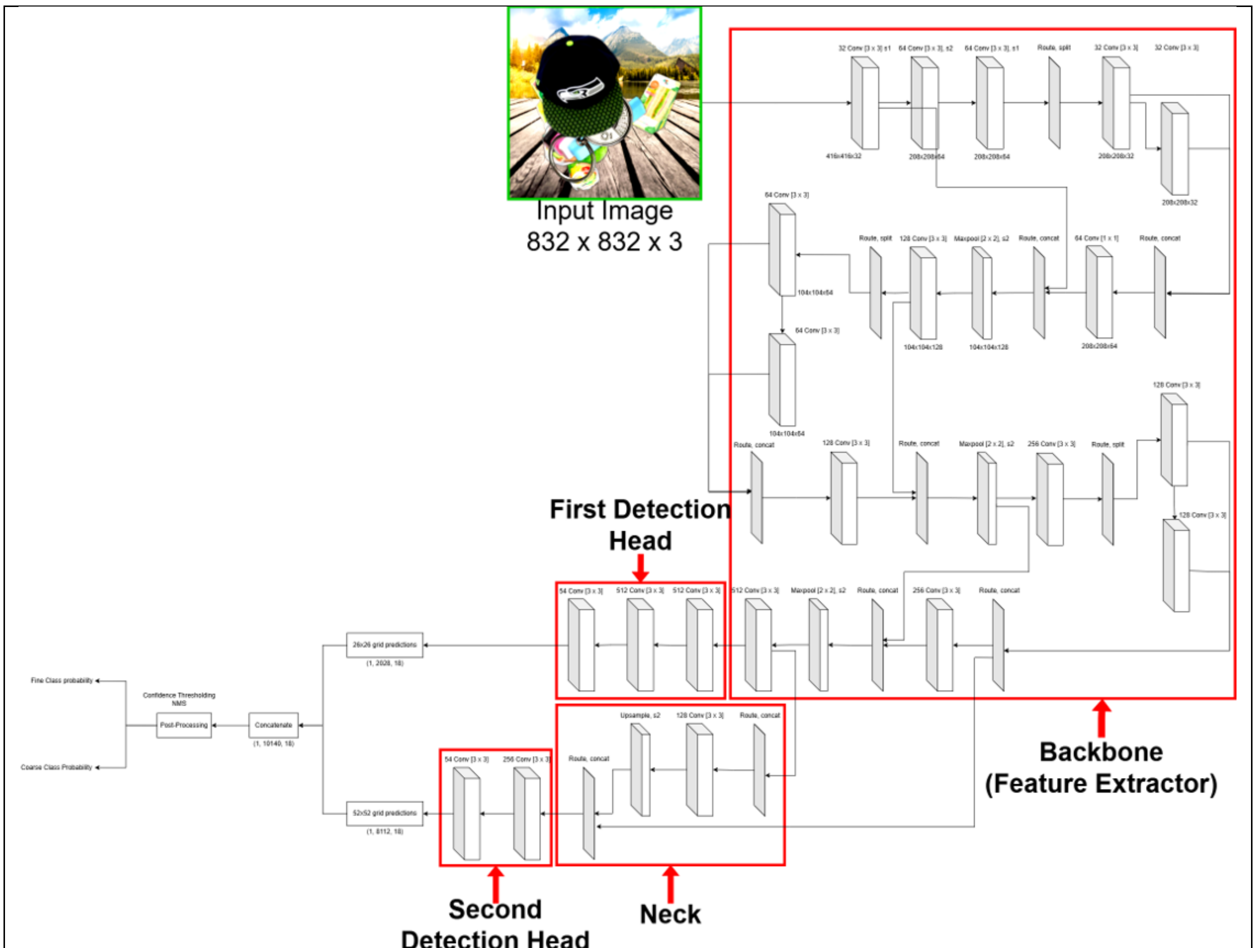


Figure 1

Figure 1 delineates the comprehensive structure of the YOLOv4-tiny model, which serves as the cornerstone of our food detection system. The network initiates with a series of convolutional layers forming the backbone, tasked with hierarchical feature extraction from the input image through iterative down sampling via maxpooling operations. These layers are fortified with batch normalization and leaky ReLU activation functions to ensure training stability and introduce non-linear transformations.

Strategic incorporation of route layers facilitates the concatenation of features extracted at varying stages of the backbone, enabling the exploitation of multi-scale representational data critical for the detection of food items across diverse size ranges.

The detection heads, composed of additional convolutional layers followed by dual YOLO layers, are engineered to predict bounding boxes and class probabilities at distinct scales. This bifurcated detection strategy empowers the model to adeptly address both diminutive and expansive objects within the visual field.

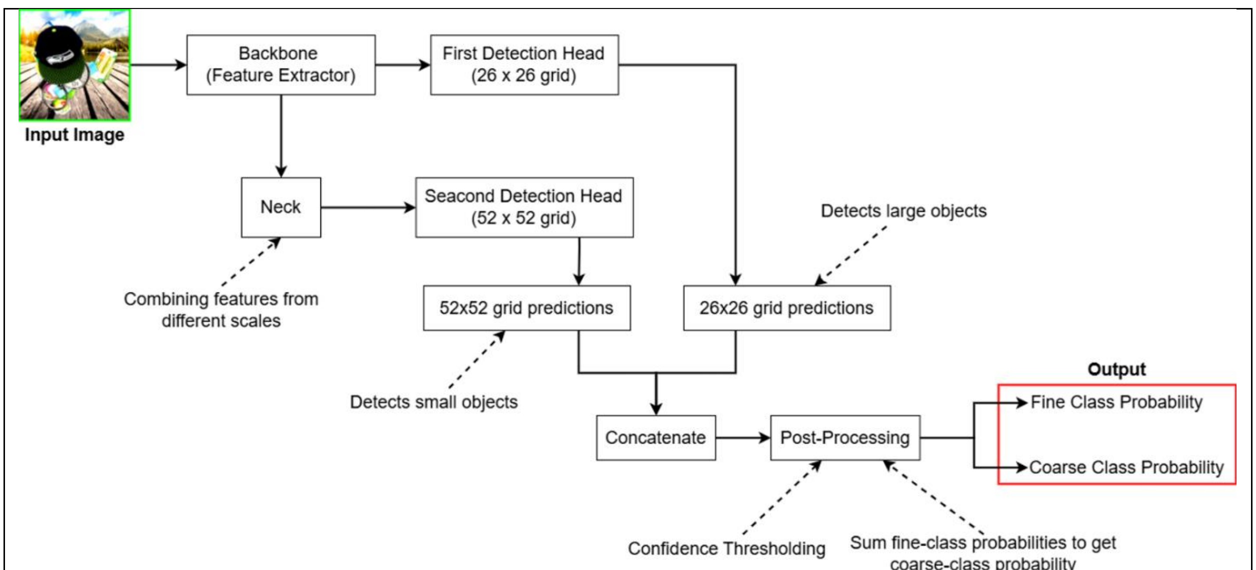


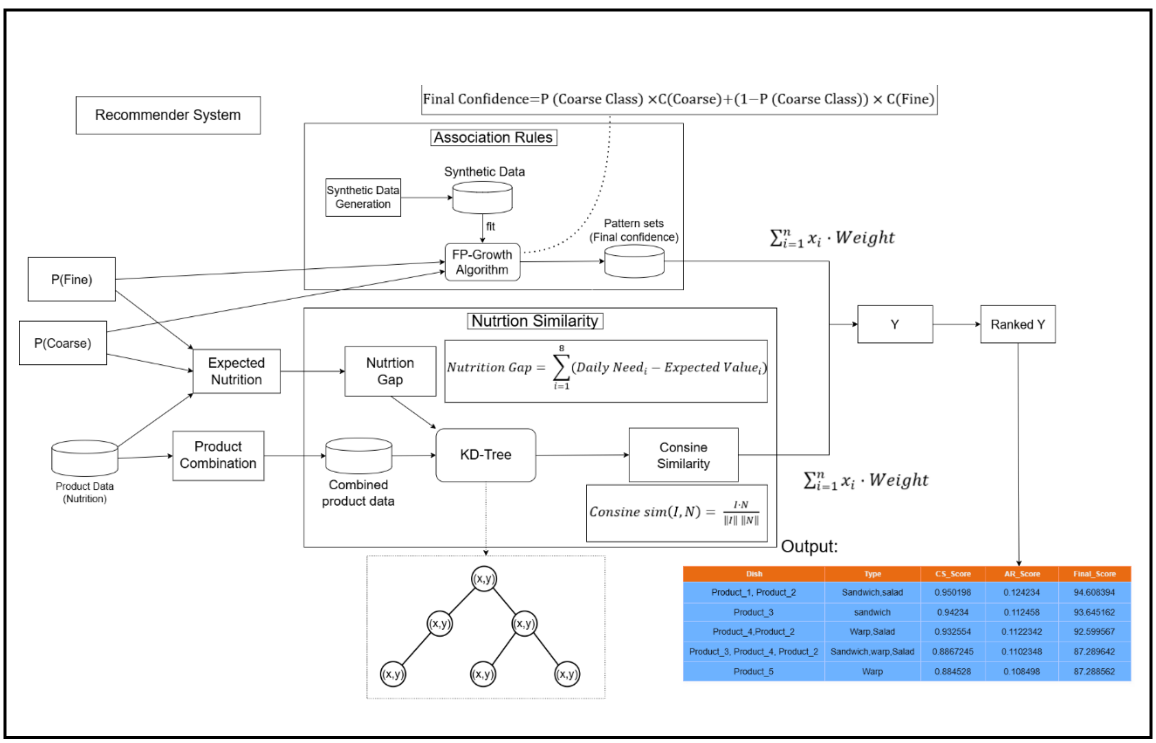
Figure2

Figure 2 elucidates the operational dynamics of the YOLO layers within the YOLOv4-tiny framework, with particular emphasis on the derivation of fine and coarse class probabilities. The model leverages two YOLO layers, each configured with anchor-based mechanisms tailored to specific size domains to optimize detection precision.

The first YOLO layer, employing larger anchor boxes, is calibrated for the identification of larger food items, yielding coarse detections.

Conversely, the second YOLO layer, utilizing smaller anchor boxes, is specialized for the detection of smaller entities, producing fine detections. For each grid cell within the feature map and for every anchor box, the YOLO layer executes bounding box regression, computes an objectness score reflecting the probability of object presence, and estimates class probabilities across 13 predefined food categories. These preliminary outputs are subsequently refined through transformation processes to yield the final bounding box coordinates and class probability estimates, which are instrumental in the identification and classification of food items within the image.

4.2.3 Recommender System



Dish	Type	CS_Score	AR_Score	Final_Score
Product_1, Product_2	Sandwich,salad	0.950198	0.124234	94.608394
Product_3	sandwich	0.94234	0.112458	93.645162
Product_4,Product_2	Warp,Salad	0.932554	0.1122342	92.599567
Product_3, Product_4, Product_2	Sandwich,warp,Salad	0.8867245	0.1102348	87.289642
Product_5	Warp	0.884528	0.108498	87.288562

For the design of our recommender system, which is specifically tailored to provide food recommendations based on input probability of Coarse class and probability of Fine class. The system is structured into two primary components: an association rule mining module and a Nutrition Similarity part.

4.2.3.1 System Component

4.2.3.1.1 Association Rule Component

The association rule component uses the FP-Growth algorithm, which is preferred over the Apriori algorithm due to its efficiency in both speed and memory usage. The FP-Growth algorithm generates association rules based on two key metrics: support and confidence.

$Support(A) = \frac{Number\ of\ transaction\ containing\ A}{Total\ number\ of\ transactions}$	This metric measures how frequently a combination of items appears in the dataset, helping us identify popular item combinations.
$Confidence(A \rightarrow B) = \frac{Support(A \cup B)}{Support(A)}$	Once we have calculated the support values, we use them to determine the confidence values, which indicate the strength of the relationship between two items. The confidence value serves as a score that contributes to our overall recommendation.
$Final\ Confidence = P(Coarse) \times C(Coarse) + \left(1 - P(Coarse)\right) \times C(Fine)$	After calculating the confidence in Fine class and Coarse Class, we use them to get Final confidence. Because we found that our synthetic data does not have particularly obvious characteristics in fine-grained categories. So, we want to add a coarse class with more obvious characteristics to calculate the final confidence.

4.2.3.1.2 Nutrition Similarity Component

The Nutrition Similarity component of our system uses cosine similarity to assess the nutrition of a food product and a profile of the user's nutritional needs. This method allows us to quantify how similar each food is based on its nutritional attributes and nutritional needs.

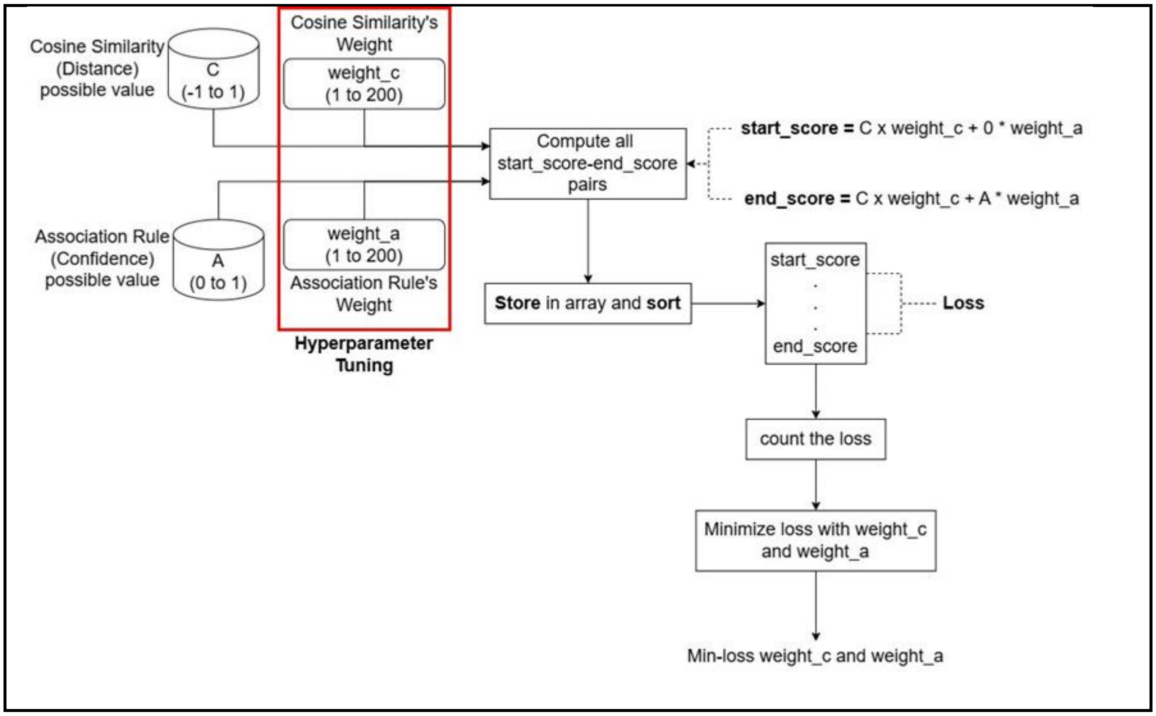
Standardize	
$\mu_j = \frac{1}{n}\sum_{i=1}^{n} x_{ij}$	x_{ij} is the value of the i-th sample in the j-th feature.
$\sigma_j = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_{ij} - \mu_j)^2}$	For each feature j, compute the standard deviation σ_j.
$z_{ij} = \frac{u_{ij}-\mu_j}{\sigma_j}$	z_{ij} is the standardized value of the i-th sample in the j-th feature.

Before calculating the cosine similarity, because the measurement units of nutritional attribute data are different and the numerical differences are large, we first standardize the nutritional data input to ensure the consistency of different types of nutritional characteristics.

Cosine Similarity	
$cosine\_similarity(x, y) = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2}\sqrt{\sum_{i=1}^{n} y_i^2}}$	x is the standardized features (nutrition of each dish in database), y is the standardized inputs (nutrition needs of user).

After standardization, to improve efficiency, we use a K-Dimensional Tree (KD-tree) instead of calculating cosine similarities for all data points. After determining the nutritional needs for a user's meal, we input these requirements into the KD-tree to find the 1,000 closest food combinations. We then compute the cosine similarity scores for these combinations, which represent their distance from the target nutritional requirements. These similarity scores are subsequently used for ranking food recommendations.

4.2.3.1.3 Hyperparameter Tuning



In our recommender system, there are two main components cosine similarity and association rules. Each component is assigned a different weight that is used to arrive at a final score that ranks the recommendations.

For hyperparameter tuning, we randomly generate cosine similarity scores within a range of -1 to 1, with increments of 0.02. Additionally, the weights for cosine similarity and association rules are randomly assigned within a range of 1 to 100. For each unique cosine similarity score, we determine the range of final scores. If there is a different cosine similarity score within this final score range, we increment the loss by 1. This process continues, allowing us to optimize the weight based on minimizing the loss. The goal is to recommend reasonable food combinations that address nutritional gaps while avoiding suggestions that may not meet users' nutritional needs.

4.2.3.1.4 Final Score Calculation

The final recommendation score is derived by combining the weighted scores from both components: the association rules scores and the Nutrition Similarity scores.

We sum these scores and rank the food items in ascending order based on their final scores, allowing us to provide personalized recommendations.

Final Score	
$$Final\ Score = (Cosine\ Similarity\ Score \times weight\_c) + (Asscciation\ Rule\ Score \times weight\_a)$$	This calculation weights the scores of both parts according to their importance, allowing us to rank foods for recommendation based on similarity.

4.3 Dataset

4.3.1 Generation image process

Frame Extraction: The code extracts frames from product videos at 0.2-second intervals using OpenCV. It processes each video file individually, saving frames as sequential JPGs in organized subfolders. This creates a structured dataset of raw product images for further cleaning and analysis.

Image Cleaning: Each frame undergoes cropping to isolate the product (800×800px) and HSV-based background removal. The system detects black backgrounds using color thresholds and replaces them with transparent or solid-color backgrounds, ensuring clean, standardized product images.

Output Generation: Processed images are saved as high-quality PNGs. The pipeline maintains consistent naming and directory structures while offering adjustable compression.

Technical Features: The solution uses HSV thresholding for background removal and offers cropping/output options. Its modular design allows easy adjustment of extraction rates, crop dimensions, and background styles, making it adaptable for diverse product imaging needs.

4.3.2 Image Synthesis

Beginning with a raw image that has gone through a generative AI model (SUDOAI) for 3D generation recorded in different rotations which are then extracted as different frames, which are pre-processed through background removal and cropping excess white space to get different perspectives of that object ; this pre-processed image is then synthesized where we select a random background and apply the transformations in random order (rotation, blur, flip, scaling and positioning; brightness and contrast adjustment) to create the dataset; during this we note that near-far constraint as well as occlusion where the front object completely occludes the one behind; We solve this by having a label validation mechanism where invalid label will be discard where one object's bounding box is completely inside the other. This is to ensure that during the synthesis object and background do not overlap.

4.3.2 Recommender System Synthetic data

For the design of our Recommender System Synthetic data, we generated synthetic transaction data to simulate eating habits. This method allows us to create a dataset that resembles similar transactions while ensuring that the resulting association rules reflect diversity.

To achieve this, we established five rules:

- No Repetition of Food Types: Within the same combination, food types will not repeat.
- Combination of Meat and Vegetables: Each combination must include at least one type of meat or seafood paired with vegetables.
- Limit on Staples: There can be no more than one staple food item, such as rice or noodles, in a combination.
- Limit on Beverages: Each combination is restricted to at most one drink.

- Exclusion of Certain Combinations: Sandwiches and wraps will not appear together in the same combination.

4.4 Evaluation Metrics

4.4.1 Object Detection

To evaluate the performance of our object detection model, we considered several key metrics that provide insights into its accuracy and effectiveness.

Model Evaluation Metrics	
$$Precision = \frac{TP}{TP + FP}$$	This metric measures the proportion of true positive detections among all positive detections.
$$Recall = \frac{TP}{TP + FN}$$	This metric measures the proportion of true positive detections among all actual objects in the dataset.
$$F1\ score = 2\frac{Precision \times Recall}{Precision + Recall}$$	This metric is the harmonic mean of precision and recall, providing a balanced measure of the model's performance.
$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i$$	This metric is a widely used metric in object detection that averages the precision across different recall levels.
 $$IoU = \frac{Area\ of\ Overlap}{Area\ of\ Union}$$ 	This metric measures the overlap between the predicted bounding box and the ground truth bounding box.

4.4.2 Recommender System Diversity

To evaluate the diversity of our recommender system, we decided to use coverage as our metric. Coverage measures the proportion of unique items recommended to users relative to the total number of available items.

Coverage	
$$Coverage = \frac{\lvert \bigcup_{u \in U} I(u) \rvert}{\lvert I \rvert}$$ $\bigcup_{u \in U} I(u)$: all unique items have been recommended to any user I : total number of items	This metric measures whether the recommender system can recommend a wide range of items, rather than repeatedly recommending the same popular choices.

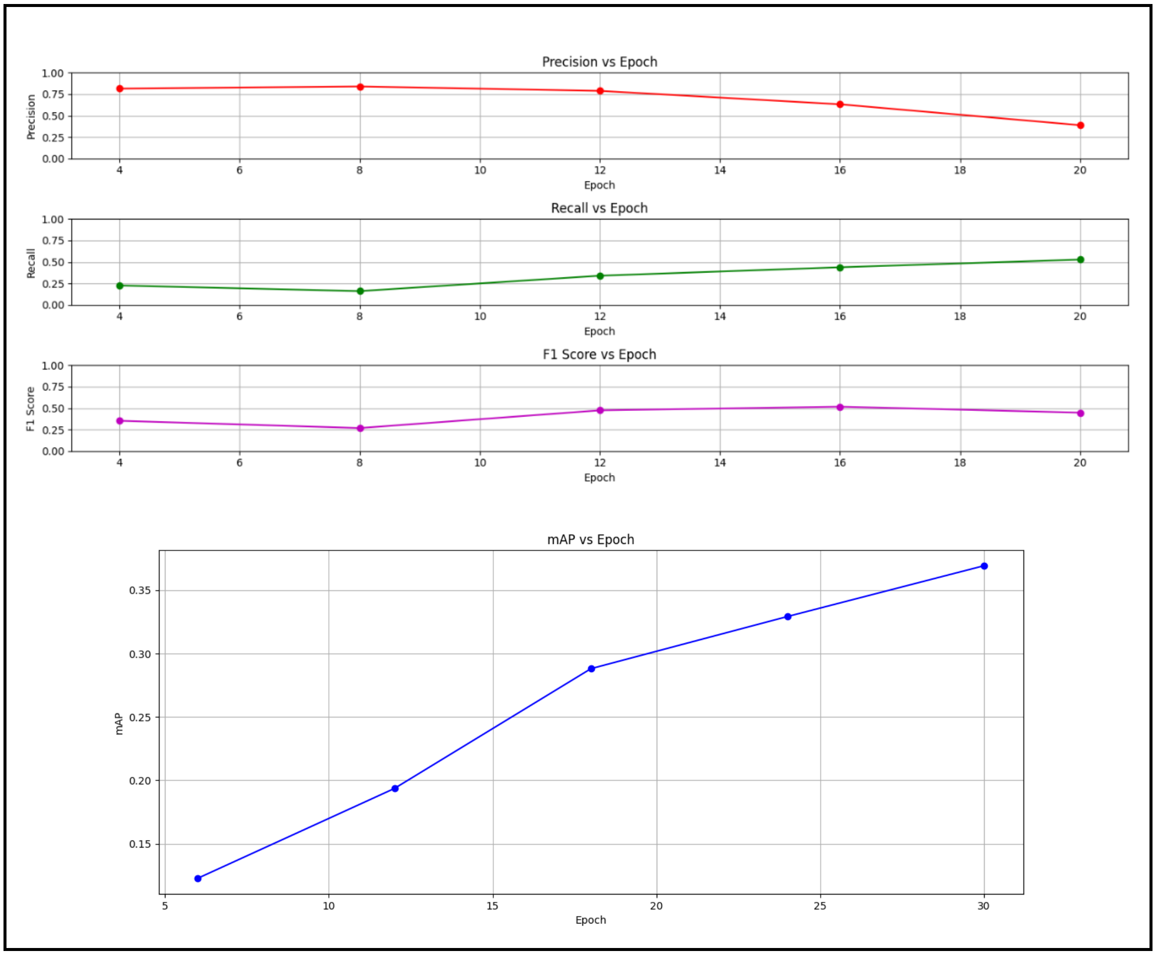
4.5 Experimental Design and Implementation Details

4.5.1 Object Detection

In experiments on object detection, we outline the experimental design for the object detection task. The goal is to evaluate the impact of various hyperparameters on the performance of the detection model. We will explore different configurations, including learning rates and Optimizer.

For our baseline configuration, we use 20,000 image data with a learning rate of 0.0001, a batch size of 8, and no layers unfrozen.

Baseline:



Here is the baseline model performance, which utilizes the baseline configuration for model training. We use this baseline model as a reference for the subsequent experiments.

Experiment Parameters:

- Learning Rate: (0.001 or 0.00001)
- Optimizer: SGD (Stochastic Gradient Descent) with Momentum

Finally, we trained three models as part of the experiment, each for an additional 20 epochs. The first model increased the learning rate to 0.001 while maintaining the baseline configuration. The second model utilized SGD (Stochastic Gradient Descent) with Momentum as the optimizer. The third model implemented a learning rate schedule.

4.5.2 Recommender System

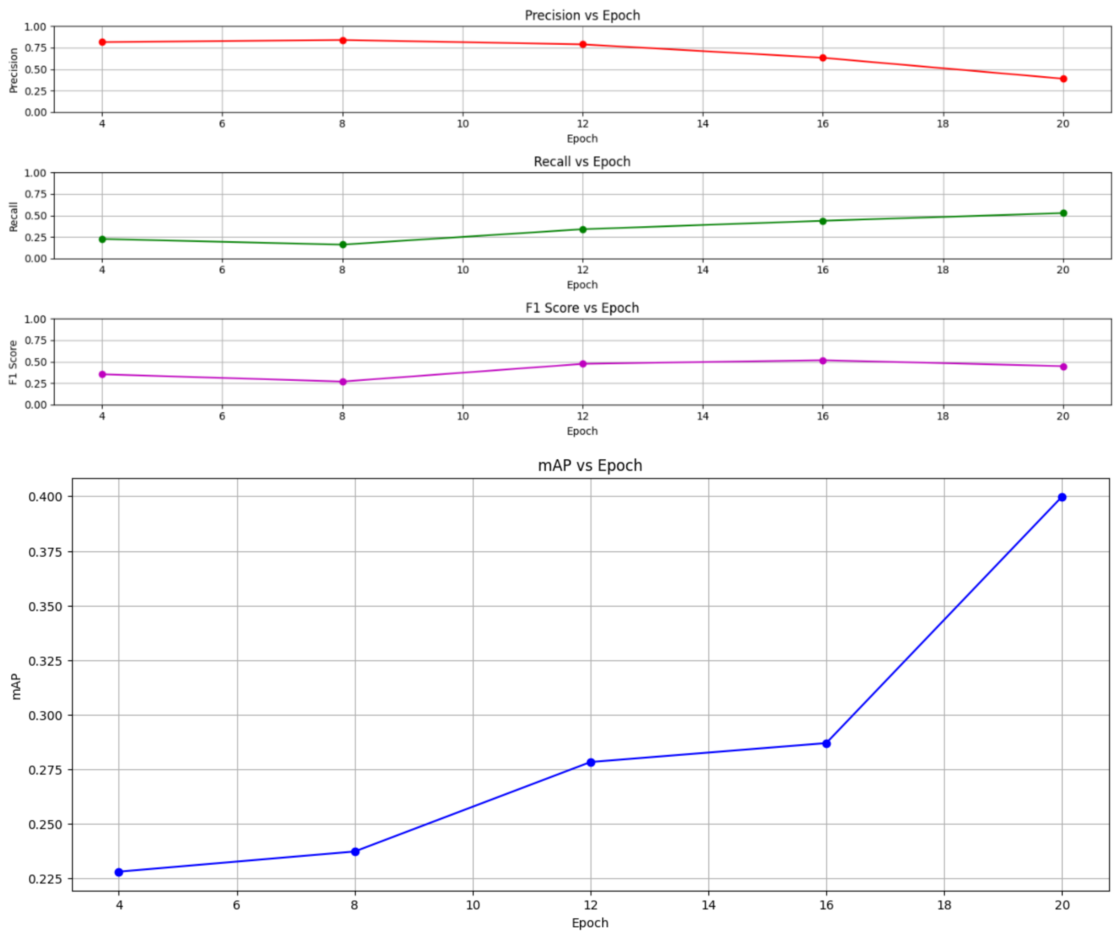
In experiments on recommender system, we conducted two main tests. The first test focused on generating synthetic data for the recommender system while varying the noise percentage. We experimented with noise levels of 0.1%, 0.5%, 1%, 5%, and 10%, maintaining a consistent total combination count of 100,000.

The second test involved evaluating the diversity of our recommender system. We used 100 synthetic images for the model's detection and then utilized the outputs from these 100 object detections to calculate the coverage.

5. Results & Analysis

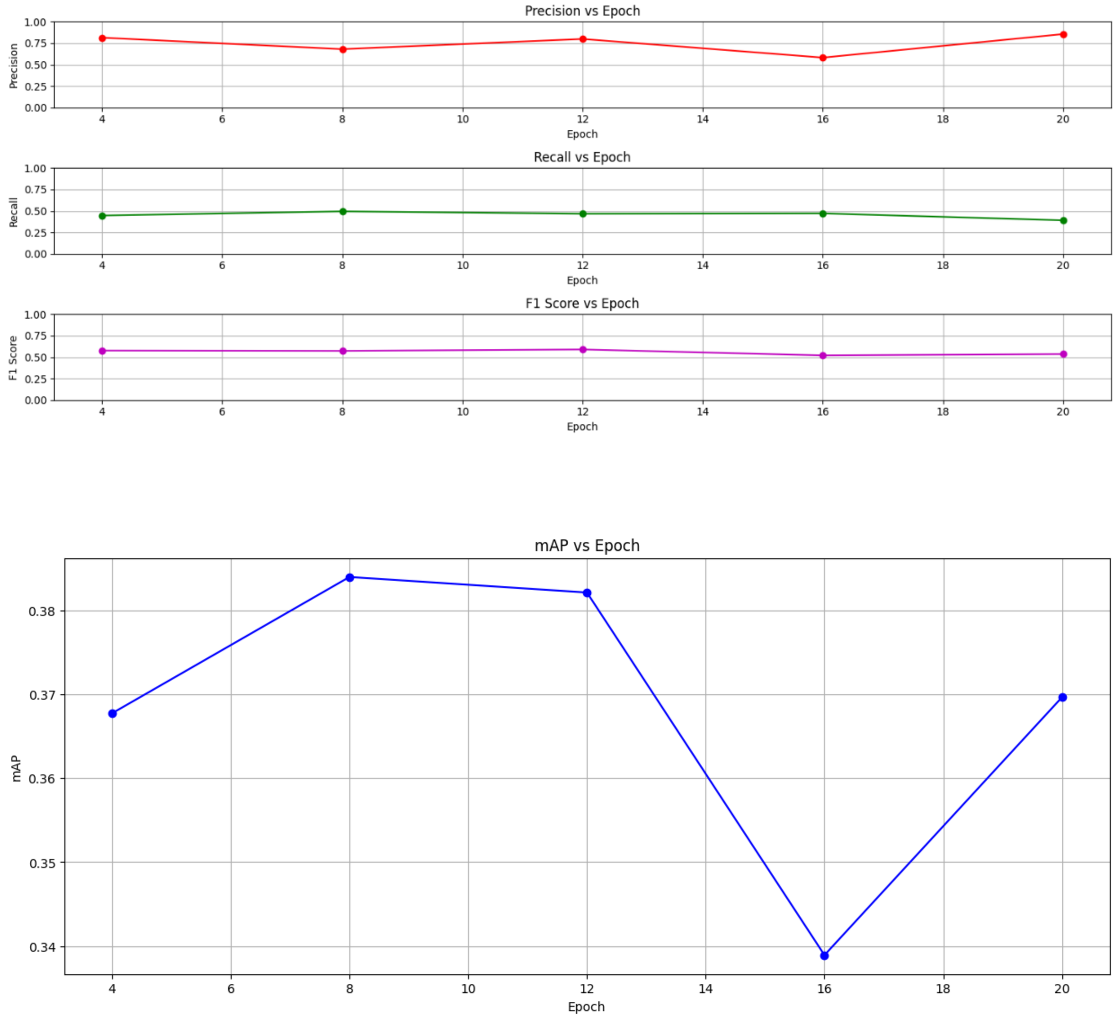
5.1. Object Detection:

In the experiments result on object detection, we decided to use F1-score, Recall and Precision and mean Average Precision(mAP) to evaluate those experiment object detection models.

Experiment 1：Learning Rate = 0.001


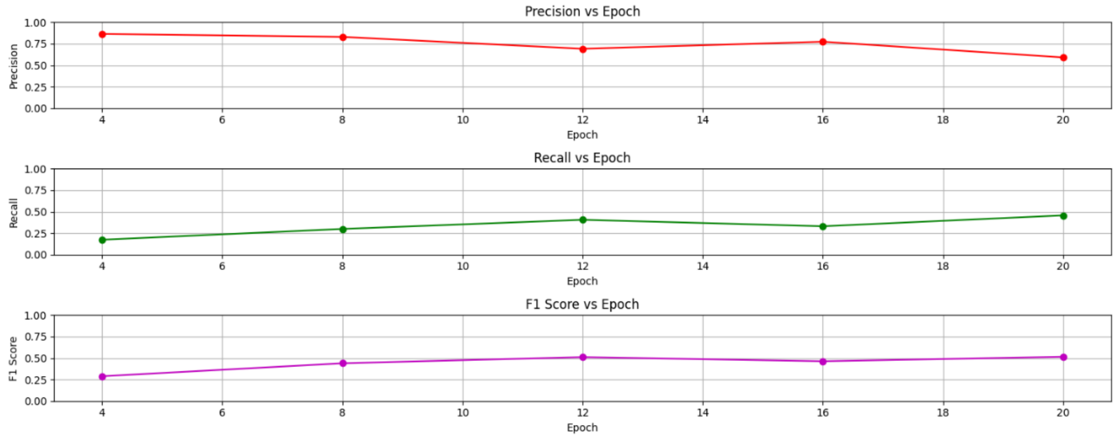
In this experiment 1, we set the learning rate to 0.001 to accelerate the model's learning process. While this adjustment has resulted in a decline in overall performance metrics such as precision, recall, and F1 score, the mAP continues to improve. This suggests that, despite the drop in some metrics, the model is becoming more effective at detecting relevant objects. We think that a learning rate of 0.001 might lead to faster convergence of the model, which could result in overfitting. While the performance during evaluation mAP improves, the model fails to generalize, causing precision and recall to drop.

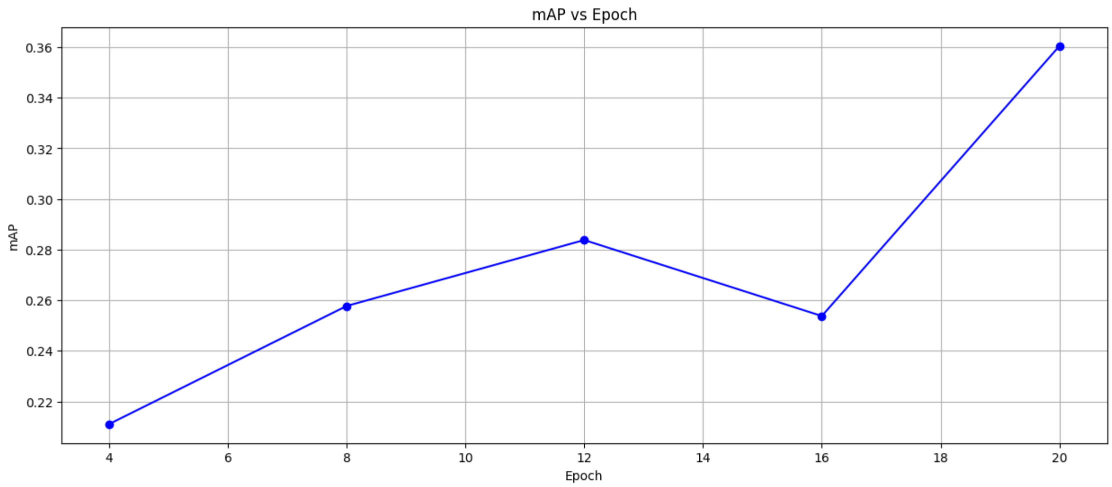
Experiment 2 : SGD (Stochastic Gradient Descent) with Momentum



In this experiment 2, we changed the optimizer to SGD to evaluate its impact on model performance. The fluctuations in precision, recall, and F1 score, along with the mAP variations, can be attributed to SGD's sensitivity to noise in the data. This sensitivity can cause significant variability in updates, leading to inconsistent performance metrics as the model learns to navigate through noisy samples.

Experiment 3：learning rate schedule

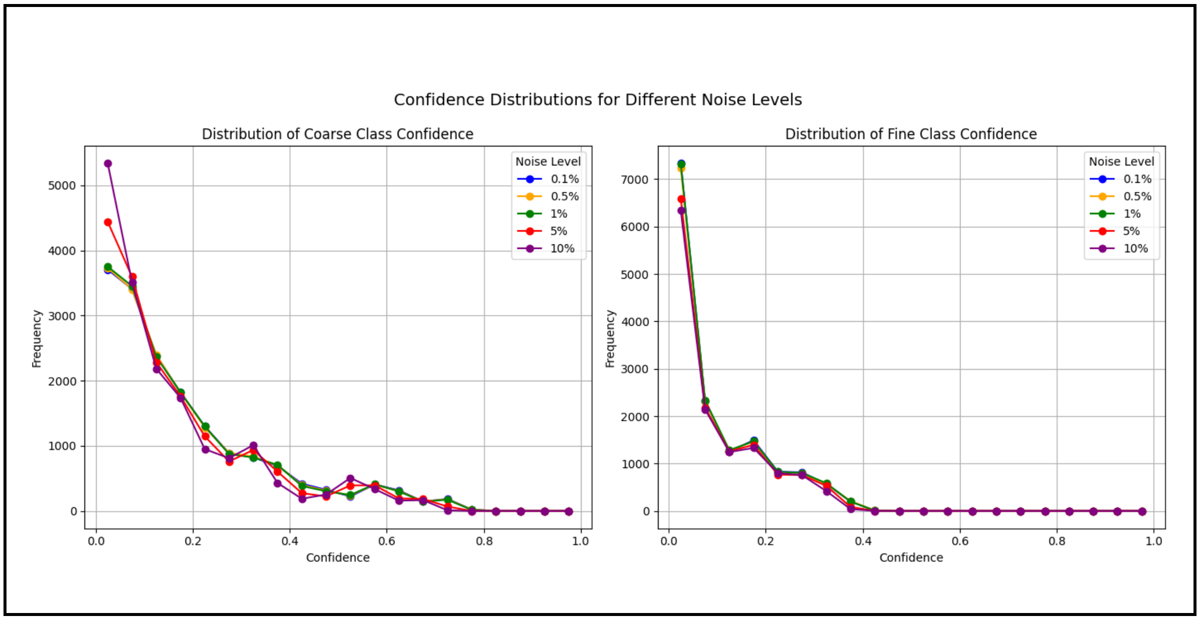




In Experiment 3, we implemented a learning rate schedule that contributed to stable precision, recall, and F1 scores throughout training. This stability arises from the gradual adjustment of the learning rate, which prevents large updates that could destabilize the model. While the mAP exhibited some fluctuations, it demonstrated an overall upward trend, reflecting continued improvement in the model's performance.

5.2. Recommender System:

5.2.1 Noise Effective of Synthetic dataset



In this line chart, we observe the distribution of confidence levels for different noise levels in our synthetic dataset. The chart is divided into two sections: one for coarse class confidence and the other for fine class confidence.

Overall, the charts can see that higher noise levels negatively affect the model's confidence. When the noise levels increase the confidence will decrease.

5.2.2 Coverage of Recommender System

To calculate the coverage of our recommender system, we used the output from 100 object detections for evaluation. The results show that our recommendation system achieves 50% coverage, meaning that only half of the food items are recommended. While this result is not ideal, we consider it acceptable given that the primary goal of our recommendation system is to suggest food based on nutritional gaps. Furthermore, the coverage is influenced by the limited number of food options available, which restricts the range of recommendations.

6. Discussion

The development and evaluation of the Object Detection of Food Nutrition Label yielded several key findings that underscore its potential to enhance dietary decision-making while highlighting areas for further refinement. These findings are derived from the integration of a synthetic dataset, real-time object detection, and a personalized recommender system, each contributing to the project's overarching goal of promoting healthier food choices.

6.1 Synthetic Dataset AI Application

The synthetic dataset, comprising 20,000 images of 15 7-SELECT products across 120 combinations, was a cornerstone of the project's AI applications, enabling robust training for object detection and recommendation systems. Generated using SUDOAI for 3D modeling, the dataset incorporated transformations such as rotation, blur, scaling, and lighting variations to mimic real-world supermarket conditions, ensuring model generalizability. Also, this generative AI approach produced diverse visual representations by simulating various angles, rotations, and perspectives, eliminating the need for labor-intensive real-world photography. By applying AI-based transformations—such as blur, scaling, brightness adjustments, and random background integration—the dataset captured real-world supermarket variability, including lighting changes and cluttered shelves, ensuring model robustness for real-time applications.

By leveraging synthetic data with these AI-driven augmentations, the project achieved a high-quality training dataset that supported the development of accurate and scalable object detection and recommendation systems tailored for supermarket product recognition and retail automation. This approach aligns with current advances in synthetic data generation, which demonstrate that carefully designed synthetic datasets can significantly improve model performance and adaptability in complex real-world applications.

6.2 Effectiveness of YOLO-Based Detection

The YOLOv4-Tiny model, implemented for object detection in this project, exhibited remarkable effectiveness in identifying 7-SELECT products within Hong Kong's 7-Eleven stores, showcasing unique attributes that distinguish it from other object detection frameworks. Its design as a lightweight, one-stage detection model enabled rapid processing of images with minimal computational overhead, achieving high precision, recall, and F1-scores when trained on a synthetic dataset of 20,000 images covering 15 products, such as the Bacon & Egg Mayo Sandwich. Unlike two-stage models like Faster RCNN, which rely on a computationally intensive region proposal network followed by classification, YOLOv4-Tiny streamlines the process by predicting bounding boxes and class probabilities in a single forward pass. This efficiency is critical for real-time applications in dynamic retail environments, where quick and accurate nutritional label detection enhances consumer decision-making.

A defining feature of the model is its innovative dual-detection framework, which separates coarse category identification (e.g., classifying an item as a sandwich) from fine-grained item recognition (e.g., identifying a Super Club Sandwich). This approach, detailed in the project's methodology, improves specificity by addressing the nuanced visual differences among 7-SELECT products, a capability less pronounced in models like YOLOv3 or YOLOv5. While YOLOv3 excels in multi-scale detection and YOLOv5 offers flexibility through PyTorch integration, both require greater computational resources, making them less suited for resource-constrained devices compared to YOLOv4-Tiny's optimized architecture. The model's ability to leverage multi-scale feature extraction, combined with a lightweight backbone inspired by CSPDarknet53, ensures robust detection of small objects, such as nutritional labels, even in cluttered supermarket settings.

The synthetic dataset's diversity, encompassing 120 combinations of lighting conditions, angles, and backgrounds, was instrumental in enhancing the model's generalizability. This dataset enabled YOLOv4-Tiny to perform effectively under varied real-world conditions, such as shelves crowded with competing products, a challenge that often confuses models trained on less diverse datasets. Hyperparameter optimization further refined performance, with experiments testing batch sizes (1 or 8), learning rates (0.0001 or 0.00001), and layer freezing strategies to balance accuracy and training efficiency. However, training on such a large dataset was computationally demanding, highlighting a trade-off between model performance and resource availability.

Despite its strengths, the model faced challenges with visually similar products, such as sandwiches with overlapping ingredients, leading to occasional misclassifications due to subtle feature differences not fully captured in the synthetic dataset. This issue underscores the model's reliance on high-quality, diverse training data to differentiate fine-grained attributes. Compared to more advanced iterations like YOLOv7 or YOLOv8, which incorporate attention mechanisms and improved small-object detection, YOLOv4-Tiny offers a more resource-efficient alternative tailored to the project's specific needs. Its ability to map detected labels directly to a nutritional database in real time, providing immediate dietary insights, further distinguishes it from generic detection frameworks, aligning with the project's goal of promoting dietary transparency.

The YOLOv4-Tiny model's effectiveness stems from its tailored design, balancing computational efficiency with high accuracy in a retail context. Its unique dual-detection strategy and optimization for resource-constrained environments set it apart from heavier, less specialized models, positioning it as an ideal solution for scalable, real-time object detection.

6.3 Personalization in the Recommender System

The hybrid recommender system developed for this project represents a sophisticated approach to delivering personalized dietary recommendations for 7-SELECT products in Hong Kong’s 7-Eleven stores, distinguished by its innovative integration of association rule mining and content-based filtering. Unlike conventional recommender systems, which often rely solely on collaborative or content-based methods, this system combines the FP-Growth algorithm for identifying item associations with cosine similarity for nutritional alignment, achieving a nuanced balance of diversity and precision tailored to user dietary needs. Trained on synthetic transaction data simulating eating habits, the system achieved a coverage of 50%, effectively recommending complementary pairings like Chef Salad and Lemon Tea to address nutritional gaps. Its personalization capabilities are uniquely designed to debunk the myth that healthy eating is expensive, leveraging accessible 7-SELECT products to support health-conscious consumers.

The system’s distinctiveness lies in its hybrid architecture, which leverages the FP-Growth algorithm to mine frequent item combinations from synthetic transaction data, using support and confidence metrics to prioritize popular and reliable pairings. Unlike the Apriori algorithm, which is computationally intensive due to multiple database scans, FP-Growth’s efficient tree-based structure enables rapid processing of large datasets, making it ideal for real-time recommendations. The content-based component employs cosine similarity, optimized with a K-Dimensional Tree (KD-tree) to compute nutritional alignment between user preferences (e.g., low-sugar or high-protein diets) and product profiles, standardizing data to account for varying measurement units. This dual approach contrasts with standalone content-based systems, which often suffer from limited diversity, or collaborative filtering systems, which face cold-start issues without extensive user interaction data.

A key feature is the incorporation of rule-based constraints within the synthetic dataset, such as prohibiting food type repetition, limiting beverages to one per combination, and mandating meat-vegetable pairings, ensuring recommendations are nutritionally balanced and contextually relevant. Hyperparameter tuning optimized the weighting of association rule and similarity scores, minimizing recommendation overlap and addressing nutritional deficiencies through a final score calculation. However, the synthetic data introduced noise, reducing confidence at higher levels, which suggests a need for refined data generation techniques. The system's moderate 50% coverage indicates underutilization of the product catalog; a limitation compared to systems with broader item exploration.

Compared to other recommender systems, such as those used in e-commerce or media streaming, this system is uniquely tailored to dietary contexts, prioritizing nutritional compatibility over general user behavior patterns. While Amazon Go's system integrates real-time object detection with recommendations, it relies heavily on user interaction data, whereas this project's synthetic data-driven approach mitigates cold-start issues in the absence of extensive user histories from the Interim Report. The hybrid design also outperforms knowledge-based systems, which are constrained by static rules, by dynamically balancing nutritional and associative factors. The system's ability to generate curated food combinations, rather than single-item suggestions, further enhances its personalization, addressing complex dietary needs.

The recommender system's effectiveness stems from its tailored hybridization and rule-based personalization, making it a robust tool for dietary guidance. Future improvements, such as integrating user feedback loops to enhance dynamic personalization and expanding the synthetic dataset for greater diversity, could further elevate its performance, ensuring broader coverage and relevance.

6.4 Other Key Findings

The project successfully demonstrated the potential of integrating YOLOv4-Tiny-based object detection with a hybrid recommender system to enhance dietary decision-making. The object detection model accurately identified 7-SELECT products, such as the Bacon & Egg Mayo Sandwich, and extracted nutritional information in real-world settings, achieving high precision and recall as evaluated by F1-scores. The hybrid recommender system, combining association rule mining (via FP-Growth) and content-based filtering (via cosine similarity), provided personalized food suggestions, such as pairing a sandwich with Chef Salad and Lemon Tea, with a coverage of 50%. This integration effectively addressed user dietary needs, debunking the myth that healthy eating is inherently expensive by leveraging accessible 7-SELECT products. The synthetic dataset of 20,000 images across 15 products and 120 combinations proved robust, enabling reliable model training under varied conditions.

6.5 Identified Challenges and Problems

The development and implementation of the hybrid recommender system for delivering personalized dietary recommendations for 7-SELECT products in Hong Kong's 7-Eleven stores presented several significant challenges.

A significant challenge in the synthetic dataset creation was ensuring data quality for the 20,000 images of 15 7-SELECT products. The use of SUDOAI for 3D modeling introduced issues with transparent packaging, which caused incorrect label annotations during image synthesis. For instance, labels fully covered by multiple items or occluded by transparent surfaces were difficult to validate, leading to the implementation of a solution to delete labels if all corners were inside another label's boundaries. However, this process was time-consuming and reduced the usable dataset size, as invalid labels had to be manually verified, impacting training efficiency.

Additionally, simulating real-world variability, such as poor lighting or crowded shelves, proved challenging, limiting the dataset's ability to fully prepare the model for diverse retail conditions.

For the YOLOv4-Tiny-based object detection system, distinguishing visually similar products posed a notable challenge. Items like sandwiches with overlapping ingredients, such as the Bacon & Egg Mayo Sandwich and Super Club Sandwich, confused the model due to subtle visual differences, resulting in occasional misclassifications. This issue was exacerbated by the synthetic dataset's limited ability to capture fine-grained features, despite its 120 combinations of angles and lighting. Furthermore, the computational requirements for training on 20,000 images were substantial, with hyperparameter tuning (e.g., batch sizes of 1 or 8, learning rates of 0.0001 or 0.00001) demanding significant resources, which constrained scalability on resource-limited devices.

The hybrid recommender system faced challenges related to synthetic transaction data quality and limited coverage. Noise in the data, introduced during generation to simulate eating habits with rules like no food type repetition, reduced the confidence of FP-Growth association rules, particularly at higher noise levels (0.1% to 10%), leading to less reliable recommendations. The system's coverage was only 36%–50% of the 7-SELECT product catalog, constrained by strict rules (e.g., limiting beverages to one per combination), which restricted recommendation diversity. Hyperparameter tuning for weighting cosine similarity and association rule scores was computationally intensive, further complicating the balance between personalization and efficiency.

These challenges reflect common issues in hybrid recommender systems, including data sparsity, cold start problems, and the difficulty of capturing nuanced user preferences and product features. Addressing them requires continued refinement of synthetic data generation to better mimic real-world complexity, enhanced modeling techniques to improve fine-grained discrimination, and optimized algorithms that balance accuracy with computational feasibility.

6.6 Future Improvements

The "Object Detection of Food Nutrition Label" project, while successful in integrating AI-driven object detection and hybrid recommendation systems, faces limitations that constrain its scalability and applicability. However, there are clear opportunities for improvement that can enhance its performance and expand its use cases.

A primary area for enhancement is expanding the synthetic dataset to increase its diversity and generalizability. Currently limited to 15 7-SELECT products with 120 combinations across 20,000 images, the dataset restricts the system's applicability to other brands and product types. Future iterations should incorporate a broader range of products, including other 7-Eleven brands and external food items, to reflect the diverse offerings in retail environments. Additionally, enhancing the dataset with more real-world variations—such as diverse lighting conditions, shelf clutter, and packaging materials—will better prepare the model for complex scenarios. Improving the image synthesis process to address transparent packaging issues, possibly through advanced 3D modeling techniques or automated label validation algorithms, will reduce annotation errors and increase usable data.

Upgrading the object detection system to YOLOv8 represents a significant opportunity to improve accuracy and robustness. YOLOv4-Tiny, while efficient, struggles with distinguishing visually similar products, such as sandwiches with overlapping ingredients, due to limited feature differentiation. YOLOv8's advanced architecture, incorporating attention mechanisms and enhanced small-object detection, offers superior performance for fine-grained identification, particularly for nutritional labels in cluttered settings. Fine-tuning YOLOv8 with an expanded dataset and optimizing hyperparameters for varied retail conditions will further mitigate misclassifications and improve real-time performance on resource-constrained devices.

The hybrid recommender system's moderate coverage and reliance on static user profiles limit its ability to deliver diverse and adaptive recommendations. Incorporating user feedback loops will enable dynamic personalization, allowing the system to adapt to evolving dietary preferences, such as low-sodium or high-protein diets. This can be achieved by integrating real-time user interaction data, similar to collaborative filtering approaches, to refine the FP-Growth and cosine similarity algorithms. Additionally, expanding the synthetic transaction dataset to include more diverse food combinations and reducing noise through improved rule-based constraints (e.g., relaxing beverage limits) will enhance coverage and recommendation variety. Optimizing the computational efficiency of hyperparameter tuning, possibly through automated machine learning techniques, will streamline the weighting of association rules and nutritional alignment scores.

7. Conclusion

This project is successfully developing an innovative proposal that combines YOLO v4 tiny-based object detection and a hybrid recommender system to provide personalized diet recommendation and has the potential to debunk the expensive equals healthy myth. The system's objective is to use computer vision techniques for food detection. It then performs food analysis and restaurant analysis. In the future, the use of YOLOv8 and broader synthetic datasets to include user feedback can improve the accuracy of detection and relevance of recommendations to better health management and health awareness.

(7445 words)

Appendix (Python Code)

Project Code
https://github.com/LutherYTT/COMPS461F_Data-Science-Project

Datasets	
Training Data	https://drive.google.com/file/d/1HwEnkcGAbojyMBAgjtRsdpl2rs51Th5U/view?usp=sharing
Validation Data	https://drive.google.com/file/d/1BzfwQlHQfW_jAZRO3ChEZ9s40h-5y5FD/view?usp=sharing
Test Data	https://drive.google.com/file/d/1Dmv40IaUDQy5ZXNWBbU7BXkPswlhlx7J/view?usp=sharing

Association Rules Mining Pseudocode

```
P1 = 0.7  # Probability that food categories do not repeat within the same combination
P2 = 0.7  # Probability that there is no more than one staple food (Rice/Noodle) in the combination
P3 = 0.8  # Probability that the combination is a pairing of meat (Meat/Seafood) + vegetables (Fruit_Cup/Salad/Vegetable)
P4 = 0.65 # Probability that there is no more than one drink (Drink) in the combination
P5 = 0.7  # Probability that Sandwich and Wrap do not appear in the same combination
noise_ratio = 0.3

categories = ['Drink', 'Meat', 'Seafood', 'Rice', 'Noodle', 'Salad', 'Fruit_cup', 'Vegetable', 'Sandwich', 'Warp']

FUNCTION GenerateCombination():
    # With probability noise_ratio, generate noise data
    IF random() < noise_ratio:
        n = randomly pick 2 to 8 according to right skewed distribution
        RETURN random selection of n categories (without any rule)

    # Generate a combination that follows the rules
    n = randomly pick 2 to 8 according to right skewed distribution
    selected_items = []

    # Because Rule 2 requires the combination to have these two items together,
    # and others cannot be placed together,
    # handle Rule 2 first to avoid insufficient remaining slots to place meat + vegetable together
    # Rule 2: With probability P2, the combination is a pairing of meat (Meat/Seafood) + vegetables (Fruit_Cup/Salad/Vegetable)
    IF random() < P2:
        meat_type = randomly pick between "Meat" and "Seafood"
```

```
        veg_type = randomly pick between "Salad", "Fruit_cup", and "Vegetable"
        ADD meat_type AND veg_type to selected_items

        # remaining is the number of remaining slots in the combination
        remaining = n - 2
    ELSE:
        remaining = n

    # If there are remaining slots in the combination, add other categories
    WHILE remaining > 0:
        available_categories = copy of all categories

        # Rule 1: With probability P1, food categories do not repeat within the same combination
        IF random() < P1:
            available_categories = categories NOT in selected_items

        # Rule 3: With probability P3, there is no more than one staple food (Rice/Noodle) in the
combination
        IF random() < P3 AND selected_items contains Rice or Noodle:
            REMOVE Rice and Noodle from available_categories

        # Rule 4: With probability P4, there is no more than one drink (Drink) in the combination
        IF random() < P4 AND "Drink" in selected_items:
            REMOVE Drink from available_categories

        # Rule 5: With probability P5, Sandwich and Wrap do not appear in the same combination
        IF random() < P5:
            IF "Sandwich" in selected_items:
                REMOVE Warp from available_categories
            ELIF "Warp" in selected_items:
                REMOVE Sandwich from available_categories

        # When no suitable food categories remain, allow duplicates
        IF available_categories is empty:
            available_categories = all categories

        # After applying the above rules, randomly choose a category from available_categories to
add to the combination
        new_item = random choice from available_categories
        ADD new_item to selected_items
        remaining -= 1

    RETURN selected_items

GENERATE 50,000 combinations using GenerateCombination()
SAVE to CSV file
```

Hyperparameter Tuning Pseudocode

```
C_VALUES = array of evenly spaced numbers from -1 to 1 (inclusive, with 0.01 increments)

FUNCTION CalculateLoss(weight_c, weight_a)
   # Calculate the minimum possible score under the current weights (based on cosine similarity)
   start_scores = array of (C_VALUES * weight_c)
   sorted_start_scores = sort the start_scores ascendingly

   FOR i = 0 TO LENGTH(sorted_start_scores) - 1:
      # Find the final score interval for the current cosine similarity score (sorted_start_scores)
      start_score_i = sorted_start_scores[i]
      end_score_i = start_score_i + weight_a

      # Find the position to insert end_score_i in the sorted_start_scores
      insertion_point = the position where end_score_i can be inserted in sorted_start_scores

      # Count how many sorted_start_scores are between insertion_point and start_score_i,
      # used as loss (each sorted_start_score corresponds to a minimum possible score under
current weights)
      loss = count how many sorted_start_scores are between end_score_i and start_score_i
   END FOR
   RETURN loss
END FUNCTION

MAX_WEIGHT_C = 200
MAX_WEIGHT_A = 200

# Initialize min_loss to a very large value for easy comparison later
min_loss = a very large value, e.g. 10000000
# Used to store the best weight combinations
best_weights = []

# Calculate loss for all possible weight combinations
FOR weight_c = 1 TO MAX_WEIGHT_C:
   FOR weight_a = 1 TO MAX_WEIGHT_A:
      current_loss = CalculateLoss(weight_c, weight_a)

      IF current_loss < min_loss:
         # Update min_loss when a smaller loss is found
         min_loss = current_loss

         # Store the current weight combination
         best_weights = [(weight_c, weight_a)]

      # Also store combinations with the same minimum loss
```

```
        ELSE IF current_loss == min_loss:
            best_weights.append((weight_c, weight_a))
        END IF
    END FOR
END FOR

PRINT "Minimum Loss: ", min_loss
PRINT "Best Weights: ", Cosine Weight, Association Weight
```

References

- Chen, M., & Liu, P. (2017). Performance evaluation of recommender systems. *International Journal of Performability Engineering*, 13(8), 1246.
- Chicaiza, J., & Valdiviezo-Diaz, P. (2021). A comprehensive survey of knowledge graph-based recommender systems: Technologies, development, and contributions. *Information*, 12(6), 232.
- Gui, S., Song, S., Qin, R., & Tang, Y. (2024). Remote sensing object detection in the deep learning era—a review. *Remote Sensing*, 16(2), 327.
- Hasan, M. R., & Ferdous, J. (2024). Dominance of AI and machine learning techniques in hybrid movie recommendation system applying text-to-number conversion and cosine similarity approaches. *Journal of Computer Science and Technology Studies*, 6(1), 94-102.
- Javed, U., Shaukat, K., Hameed, I. A., Iqbal, F., Alam, T. M., & Luo, S. (2021). A review of content-based and context-based recommendation systems. *International Journal of Emerging Technologies in Learning (iJET)*, 16(3), 274-306.
- Jiang, Z., Zhao, L., Li, S., & Jia, Y. (2020). Real-time object detection method based on improved YOLOv4-tiny. *arXiv preprint arXiv:2011.04244*.
- Kundu, R. (2022). F1 Score in Machine Learning: Intro & Calculation. V7. https://www.v7labs.com/blog/f1-score-guide
- Linder-Norén, E. (2022, April 21). PyTorch-YOLOv3. GitHub. https://github.com/eriklindernoren/PyTorch-YOLOv3
- Papadakis, H., Papagrigoriou, A., Panagiotakis, C., Kosmas, E., & Fragopoulou, P. (2022). Collaborative filtering recommender systems taxonomy. *Knowledge and Information Systems*, 64(1), 35-74.

- QuakeLogic. (2024). Understanding Signal-to-Noise Ratio (SNR) and Its Importance in Seismic and Structural Health Monitoring - QuakeLogic. https://blog.quakelogic.net/understanding-signal-to-noise-ratio-snr-and-its-importance-in-seismic-and-structural-health-monitoring/
- 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* (pp. 779-788).
- Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster R-CNN: Towards real-time object detection with region proposal networks. *Advances in Neural Information Processing Systems*, 28.
- Ricci, F., Rokach, L., & Shapira, B. (2021). Recommender systems: Techniques, applications, and challenges. *Recommender Systems Handbook*, 1-35.
- Sarma, D., Mittra, T., & Hossain, M. S. (2021). Personalized book recommendation system using machine learning algorithm. *International Journal of Advanced Computer Science and Applications*, 12(1).
- Shah, D. (2022). Mean Average Precision (mAP) Explained: Everything You Need to Know. https://www.v7labs.com/blog/mean-average-precision
- Song, H., Zhang, H., & Xing, Z. (2021, July). Research on personalized recommendation system based on association rules. In *Journal of Physics: Conference Series* (Vol. 1961, No. 1, p. 012027). IOP Publishing.
- Wang, L., Zhou, K., Chu, A., Wang, G., & Wang, L. (2021). An improved light-weight traffic sign recognition algorithm based on YOLOv4-tiny. *IEEE Access*, 9, 124963-124971.

- NielsenIQ. (2024). NIQ unveils comprehensive 2024 Consumer Outlook Report: Navigating global trends and insights for strategic growth. https://nielseniq.com/global/en/news-center/2024/niq-unveils-comprehensive-2024-consumer-outlook-report-navigating-global-trends-and-insights-for-strategic-growth/