AI-Augmented Food Analyzer for Visually Impaired Persons

By

TALHA ALI ASHFAQ
Enrollment No. 01-133212-165

TAHA BIN TARIQ
Enrollment No. 01-133222-076

Supervised By

ENGR. MUDASIR WAHAB

Session 2025-2026

A Report is submitted to the Department of Electrical Engineering,
Bahria University, Islamabad.
In partial fulfillment of requirement for the degree of BS(EE).

Certificate

We accept the work contained in this report as a confirmation to the required standard for the partial fulfillment of the degree of BS(EE).

_____.	_____
Head of Department	Supervisor
_____.	_____
Internal Examiner	External Examiner

Dedication

Dedicated to our parents.

Acknowledgments

All praise is due to the Almighty, whose mercy and guidance made this project possible. I would like to express my deepest gratitude to my supervisor, whose continuous support, constructive feedback, and insightful guidance helped shape this work from its earliest concept to its final implementation. Their expertise and patience played a vital role in directing the research and development process.

I am sincerely thankful to my department and university for providing the opportunity, resources, and academic environment necessary for undertaking this project. The encouragement from faculty members and peers has contributed significantly to both the technical and intellectual aspects of this work.

Special appreciation goes to my family and friends for their understanding, moral support, and constant motivation throughout the challenges of this journey. Their presence made even the most difficult moments manageable. Lastly, I extend gratitude to the global research community whose contributions in the fields of deep learning, computer vision, and assistive technologies laid the foundation upon which this project was built.

Abstract

Visually impaired individuals face significant challenges in identifying food items and understanding their nutritional content, often relying on others for assistance in daily dietary decisions. This dependency not only limits personal independence but also increases the risk of consuming inappropriate or unhealthy meals. To address this gap, this project presents an AI-Augmented Food Analyzer, a mobile application designed to identify food items and their calorie content using image based analysis. The system utilizes state of the art deep learning models, including ResNet101, EfficientNet-B4, and ConvNeXt-Small, trained on a saliency enhanced dataset to improve classification accuracy and model focus. Through this approach, the models achieved an accuracy exceeding 90%, ensuring reliable performance in real world food recognition scenarios.

The application leverages visual saliency to highlight key regions within an image, enabling more robust feature extraction despite challenges such as inconsistent plating, varied lighting conditions, and mixed food compositions. Once the food item is identified, its calorie information is estimated and communicated to the user through an accessible interface designed specifically for visually impaired individuals. Features such as audio prompts, simplified navigation, and nonvisual feedback ensure ease of use and independence. By integrating advanced deep learning techniques with a user centered design philosophy, this project provides a meaningful solution that enhances accessibility, supports healthier dietary choices, and contributes to the growing field of AI driven assistive technologies.

Contents

List of Figures

Chapter 1

Introduction

1.1 Project Background

Human beings interact with food first through sight and then through taste. The visual perception of food determines freshness, appeal, portion size, and even safety. A person can instantly recognize whether a plate contains rice or pasta, whether a dessert is chocolate or red velvet, or whether a meal appears healthy or excessive. This ability is so natural that most people never realize how dependent they are on vision while eating. For visually impaired individuals, however, this natural process is interrupted. A simple act such as identifying what is placed on a plate becomes dependent on external assistance.

According to international health statistics, millions of individuals worldwide suffer from partial or complete visual impairment, affecting their ability to independently perform daily life activities [1]. While assistive technologies have significantly improved mobility and digital accessibility through screen readers and speech based systems, dietary independence remains an underdeveloped area.A person with visual impairment will often depend on their sense of smell and touch to help them identify their food type; however, due to limitations of these senses, they cannot determine exactly what ingredients are in a particular dish, its allergens, or how many calories are present. As we become more and more aware of nutrition in regards to the increases in diabetes, cardiovascular disease, obesity, and food sensitivities diseases, there is a direct correlation between the inability to assess food independently and the associated health risks and diminished self esteem associated with food.

The evolving field of AI has been instrumental in improving how machines are able to understand their environment through images. In the past, computer vision relied heavily on artificial and constructed feature

sets such as SIFT and HOG, which had poor performance depending on how light or complex the background was [2].With deep convolutional neural networks becoming mainstream, machines are now capable of learning features from images by using the image data itself to build hierarchical features [3].Residual Learning was used in architecture solutions like ResNet to allow the use of very deep networks without the gradient degradation that can happen with regular networks [4].EfficientNet used an optimized scaling method to balance performance improvement with computation for the amount of additional depth, width, and resolution of a given architecture. ConvNeXt modernized convolutional networks by using design principles based on transformer architectures to improve the design of the convolutional network while also keeping the efficiency of Convolutions.MobileNetV3Large made Mobile architectures appropriate to be deployed on mobile devices low power requirement while still allowing for enough computation in order to perform the mobile tasks.

Food recognition is still a very complicated task with many different things to consider; deformation, lack of set structure, and visual differences, can make food recognition difficult. Food items will present with very different colours, styles of presentation, textures etc., based on how they were prepared, the environmental light they are encountered with, and many other variables to define their appearance. Visually similar food items can be fundamentally different food items and belong to different categories. Attention mechanisms have been put forward as potential ways of guiding neural networks towards more informative spatial and channel locations through the use of Convolutional Block Attention Modules (CBAM). There are also many approaches to isolating the food from the background noise plates, forks, tables etc [5]. using saliency detection, for example; Graph Based Visual Saliency, UNet based saliency extraction,

region contrast saliency, and DeepLabV3+ semantic segmentation.

The project name is "AI Augmented Food Analyzer for People Who Are Blind" and it is a combination of several of these advanced ideas and implementations into an integrated support framework i.e. An Assistive Technology Framework. The AI system will make use of pretrained models like ResNet101, EfficientNetB4, ConvNeXt Modern, ConvNeXt Small, EfficientNetV2 and MobileNetV3Large to compare their use. The Food101 dataset is used as the primary dataset for training the food classification system and the COCO dataset provides negative nonfood images in order to help reduce false positives and properly identify food in its background. Finally, YOLOv8 has been used to train the multiple food detection scenarios by applying augmentation techniques to enhance the robustness of the food detection when found in a single frame or image.While the rationale for this project is not simply an academic exploration of deep learning architectures, it is also about converting artificial intelligence into meaningful assistance. After the food has been detected, the user can hear the name of the food, how many calories it contains and other related food ingredient types through speech output. By hearing a food item identified through a voice synthesizer, a visually impaired person can thereby independently know what item they are about to eat. Therefore, the project represents a combination of technology and social responsibility to assist in rebuilding a person's confidence and self reliance through intelligent perception.

1.2 Problem Description

The topic of assistive food recognition systems for visually impaired people has only seen limited research compared with the larger area of object recognition image classification. Most commercial food recognition systems

have been designed to be used by a typical sighted person and, therefore, utilize visual confirmation of the image through graphical user interfaces GUIs. Thus, they are based on the assumption that the user will be able to view the photographed image in order to validate the classification of it. For many users this is not true, resulting in a significant usability issue related to the primary purpose of these systems food classification. In addition, food classification has technical difficulties that result from two major challenges. The first challenge is intra class variation in which one food category can be seen in multiple ways due to cooking variation how food is prepared, garnishing how food is presented, camera angle where the photograph was taken, and illumination how brightly the food is lit. Thus, a given food will look different, based on the cooking method used to prepare it, the manner in which it was presented, the camera angle that was used to take the photo, and the amount of brightness of the light that illuminated it. The second challenge associated with food classification is inter class similarity where two different food items appear visually similar to each other. For instance, under certain light levels hummus has a similar look to mashed potatoes; thus, they are difficult to classify as a result of this graphical commonality between their respective classes.

Another large concern of image classification involves multiple food types included within the same image. Classification networks typically provide one primary label for each image and do not develop associated bounding boxes. In real life settings, dining includes many plates containing multiple food types. YOLOv8 as a new and improved object detection tool can detect and define each food type simultaneously in one image.

Recognition of food is made more difficult by background interference. For example, visual distractions from elements that aren't food, like cutlery, table cloth patterns, and human hands can confuse classification net-

works. To help reduce false detections, we include negative samples from the COCO dataset during training to teach the model to differentiate between food and nonfood objects. In addition, saliency based techniques and segmentation models e.g., UNetSaliency and DeepLabV3+ are used to highlight areas of interest and suppress visual distractions from the background. Lack of an accessible food analysis system, in addition to technical difficulties, can prevent some users from being able to independently identify food items. Thus, many users who are blind may unintentionally ingest food items that contain allergens or excessive calories. People who rely on others for determining their dietary choices often have lower self esteem, feel a loss of autonomy, and have little confidence in their ability to live on their own. Therefore, the primary goal of this project is to create an accurate, reliable, and accessible AI powered food analyser that will give meaningful input to visually impaired users in real world environments.

1.3 Project Objectives

This project aims to create an app for mobile devices that can recognize what type of food it sees in real time with a camera. The app will capture a picture of food in front of someone using their phone and then analyze it through deep learning algorithms to identify the category of the food they captured. To accomplish this goal, we will take a simple camera like a smartphone camera and turn it into an intelligent assistant that can help identify food without any manual input from the user.

Along with recognizing the food item, the app will provide details about the specific food item regarding its main ingredients, approximate calorie count, etc. By providing users with ingredient information, we can educate them about what they are consuming and allow them to better track how

many calories they have eaten to make healthier eating decisions.

While estimating a calorie count is not an exact science because it uses pre created nutrition databases, it can still offer a reasonable indicator of the nutrition a person should be consuming, particularly if a person has an illness or other dietary restriction.

The project's main objective is to create an easy to use and user friendly interface that provides real time results to the user in a way that they can easily access. Because this system was designed mainly for visually impaired users, the interface will have as few complications and provide as much clarity as possible. Because the output structure is set up to provide auditory feedback without any need for the user to look at their screen, this auditory feedback will guarantee the user receives the information they are looking for. In addition to its intended use as a resource for people with visual impairments, this application will ensure that the user experiences smooth, fast, and easy to navigate interactions with the technology so that the technology can act as an aide and not as a tool. The project will combine intelligent image recognition, nutrition awareness, and assistive design into one complete solution.

1.4 Project Scope

The goal of this project is to build a food recognition and analysis system by developing it primarily with the Food101 dataset and strengthening it with negative COCO images to improve the ability to differentiate food from the background. The system consists of a classification component based on deep convolutional networks for food recognition and a detection component based on YOLOv8 to locate food in the image. Saliency and segmentation techniques will be used to help refine the area of focus and

reduce classification mistakes due to background clutter. Image augmentation techniques will be used to model real world image variability and increase the ability of the system to generalize.

The system is aimed at supporting the visually impaired by providing instant audible feedback about what food is present, what ingredients it contains, and an approximate number of calories. While the current implementation supports only previously defined food items and categories contained in the dataset, due to the flexibility of the architecture, it can be expanded in the future to include different types, types of foods, and regional versions of each food. The project does not deliver an exact volumetric estimation of caloric content through depth sensing; this requires building a three dimensional model; rather, it provides approximate estimates of nutritional values using a standard reference. Within these boundaries, the project creates the foundation for an AI driven assistive food recognition system to be further developed in future research and development.

Chapter 2

Literature Review

Technological innovation is never isolated. Each intelligent system reflects years of theoretical research, algorithmic advancement, and experimental validation; therefore, an AI Augmented Food Analyzer for the visually impaired must be based on a thorough understanding of advancements in the following areas: deep learning, classification of food images, attention mechanisms, saliency modelling, object detection frameworks, and designing assistive interfaces. This chapter will discuss each of these fields in order to establish a fundamental basis for the suggested AI Augmented Food Analyzer.

2.1 Deep Learning for Image Classification

Deep Learning has driven a revolution in the Computer Vision Field over the past 10 years. Before Deep Networks were widely adopted, the majority of image classification systems utilized handcrafted techniques to extract features from images e.g., SIFT, HOG. The majority of conventional image classification systems required researchers to manually design visual descriptors for edge, gradient, corner, and texture extraction from the images before they could perform classification. While these methods performed moderately well in constrained environments, they had significant difficulties dealing with variations in real world images caused by changes in illumination, viewpoint, scale, and scene complexity. These limitations were particularly evident in fine grained visual domains such as food recognition, where object appearance is often highly deformable and variable.

Deep Convolutional Neural Networks CNN changed the way we approach visual recognition by making it possible to learn from raw pixel images and automatically generate hierarchical feature descriptors with-

out any human intervention. With the help of the CNN framework an automatic learning process takes place where low level features are extracted early in the learning, medium level features are extracted in the middle of learning, and high level features are extracted at the end of learning. Because hierarchical learning produces richly structured features with highly discriminative abilities there have been significant improvements in performance over handcrafted features on Image Classification. The fundamental operation of convolution used in CNNs is mathematically expressed as shown in Equation 2.1.

$$F(i,j) = \sum_{m} \sum_{n} I(i-m, j-n) \cdot K(m,n) \tag{2.1}$$

where:

- $F(i,j)$ is the output feature map
- $I(i-m, j-n)$ is the input image
- $K(m,n)$ is the convolution kernel
- m, n are kernel indices

Researchers increased the number of layers in networks to improve the performance of convolutional neural networks, known as convolutional neural networks. Adding layers causes issues with optimizations such as vanishing gradients and degraded performance. To solve this problem, He et al. developed the Residual Network ResNet architecture, which uses residual learning to allow deeper networks to improve upon the performance of prior deep learning networks. The ResNet architecture redefined the deep learning problem of learning a complete transformation as the problem of learning residual mappings from the input to the output by creating skip

connections shortcut connections to allow the residual image to skip over intermediate feature maps and combine directly with the feature maps of the output layer. By maintaining the gradient path during backpropagation through the use of residual connections, residual connections enabled successful training of very deep networks like ResNet101 without experiencing degradation of optimization performance. The concept of residual learning can be represented mathematically as shown in Equation 2.2.

$$H(x) = F(x) + x \tag{2.2}$$

where:

- $H(x)$ is the residual output
- $F(x)$ is the learned mapping
- x is the input feature

ResNet's attributes exceed merely depth, and, for instance, in food recognition tasks, there are many visually similar foods that only vary in minor texture, shape, or arrangement of the ingredients variance. ResNet101's deep hierarchical feature extraction capacity allows the model to identify these subtle differences more accurately than a model with less depth; consequently, ResNet is one of the most commonly used backbone models for food recognition, as well as object detection and segmentation, and medical image analysis.

Although ResNet showed the viability of building deeper models, researchers quickly realized that just making a model deeper didn't lead to achieving greater efficiency. Many of the other models that researchers were experimenting with became far too large and computationally expensive; therefore, to help combat this issue, Tan and Le introduced a new model

called EfficientNet, which they created by scaling the network depth, width and input resolution using a new proposed compound coefficient in order to create a family of EfficientNet models that provide better accuracy with less number of parameters and significantly lower computational costs.

EfficientNets have been popular because they are the most accurate and efficient of all deep learning models. This is very important for food recognition systems since they will operate in mobile or embedded settings when deployed.EfficientNet architectures allow for the same or greater predictive power while having less memory usage and inference time. This makes them ideal for real time food recognition applications

As the field moved forward, some researchers revisited how they designed convolutional neural networks. While transformer based vision models gained much interest, Liu et al. found that convolutional architectures still had a lot of untapped potential if modernized properly. They presented ConvNeXt, which redesigned standard ResNet architectures with architectural improvements derived from transformer based models while keeping convolutional operations intact. New features of the ConvNeXt architecture included larger kernel sizes, inverted bottleneck structures, layer normalization, and optimized block designs; all of which resulted in better representation power and improved efficiency during training [6].

ConvNeXt's importance in the food recognition domain is largely based on its ability to model larger scale contextual relationships within an image versus local rigid patterns. Most food items have very subtle global structural patterns versus a rigid local pattern, therefore the overall larger receptive fields and contextual modeling found within ConvNeXt help distinguish similar looking dishes. Both ConvNeXt Small and ConvNeXt Modern are capable of being fine tuned modern backbone networks for visually classifying finely detailed objects.

In addition to large scale models, smaller lightweight deep learning networks have become more prevalent for use on mobile and edge deployments. Computational resources are limited in many assistive technology applications, including mobile food recognition systems, so low latency inference must be provided. To fulfill this need, Howard et al. created a highly optimized lightweight convolutional network specifically designed for mobile devices called MobileNetV3. MobileNetV3 uses several different methods, including depth wise separable convolutions, squeeze and excitation attention modules, efficient bottleneck blocks, and hardware aware neural architecture search to achieve maximum performance under very strict computational limits [7].

MobileNetV3Large gives a great tradeoff between prediction accuracy and performance. Unlike the comparable larger architectures of ResNet and EfficientNet, MobileNetV3 does not have to make a compromise in classification accuracy but performs much better in terms of computational costs than those architectures allowing for its increased use with assistive food recognition mobile applications where computational accuracy and battery consumption are critical elements in the design of the solution.

Together, these advancements in how deep learning technologies have evolved through the various deep learning architectures from residual learning to enable deeper networks to compound scaling for better efficiencies to modernized convolution methodologies that further allowed for greater contextual representation and lighter architectures for easier mobile deployment have allowed for addressing many of the constraints experienced by earlier generation architectures thereby providing many exciting capabilities for image classification.

For the purposes of this project, the aforementioned architectures will be used as the basis for evaluating the comparative usage of the pretrained

deep learning models on image classification of food. These include the use of ResNet101, EfficientNetB4, EfficientNetV2, ConvNeXt variants, and MobileNetV3Large to identify which architecture can provide the best balance of classification accuracy, computational efficiency, and suitability for deployment for assistive food recognition applications.

2.2 Food Recognition and Benchmark Datasets

Food recognition is very challenging to classify in terms of images because food items have so much variance and are non rigid. Food has a lot of variability when compared with rigid objects such as cars, furniture, and tools; food does not have a well defined structure like those types of rigid objects, and food can also vary greatly in shape depending on how the food was cooked, what type of food is being cooked, the layout in which the food is arranged, how the food is displayed culturally, how the food is photographed, and how bright the light is that is being used to photograph the food. One category of food can have a wide range of visual characteristics at different samples; food that is completely different can appear to be similar visually if the colours, textures, or garnishes overlap. These elements contribute to the fine grained classification problem of food recognition and require highly discriminative feature extraction and the design of robust datasets in order to develop these systems successfully [8].

The other factor that affects how successful deep learning based food recognition systems are would be the neural network architecture being used; however, the most critical factor in developing these systems would be the quality, diversity, and representativeness of the datasets used to train and test the neural network. In the area of computer vision, food recognition systems have a need for benchmark datasets to serve as a

foundation for the recognition model to learn visual patterns, generalise to unseen instances, and to compare to other recognition models. For food recognition systems, benchmark datasets have an even more essential role because of the variety of food imagery; food recognition systems must experience many different types of food in order to be able to train on highly diverse examples of realistic food.

The Food101 dataset is one of the most important benchmark datasets for research on food recognition. Bossard and colleagues created this dataset, in order to provide a large scale dataset for food image classification. The dataset contains a total of 101 unique food categories, with 1,000 images per food category, providing a combined 101,000 images of food. The large scale of the dataset provides enough data for training deep neural networks. Furthermore, there is a balanced number of samples from each category contained in the Food101 dataset.

A major strength of the Food101 dataset is the realistic distribution of images used in the dataset. The Food101 dataset includes food images that have been taken from real world sources as opposed to laboratory curated datasets, which contain segmented and or perfectly centered food images. As a result, the Food101 dataset has significant variations between images with respect to their background clutter, lighting conditions; plating styles; camera view positions; occluded foods; and image quality. This realism gives the Food101 dataset much greater usefulness in reflecting the challenges faced by real world food recognition systems that are actually deployed in restaurants, homes, cafeterias and assistive mobile applications.

The intentionally imperfect design of the Food101 dataset is another important aspect of this dataset's characteristics. The authors of the dataset included a certain amount of noise in the labels and inconsistency in the

annotations in order to enable the training of models on data that is more representative of the real world. Some of the images in the dataset may have weak labels, with ambiguously presented food items or some incorrectly labeled items. While this makes the task of classifying the images more difficult, it also encourages the creation of more robust classifiers, which will generalize better to data that does not conform to realized ideal conditions in a database. As such, Food101 should not only be regarded as a method for evaluating classification performance, but also as a pragmatic way of assessing the robustness of classifiers developed using the dataset against reality.

However, the Food101 dataset alone is not enough for developing food recognition systems for real world use. A system that only trains food classifiers based on food will classify a large number of non food objects that are originally random objects as foods when fed into a real world camera. In most cases, when the system is actually used in an assistive mobile application, the input may include images that have no food at all or include objects that do not relate to any food. If the classifier is never trained on any negative examples, then it will produce false positive results, leading to reduced dependability and user confidence.

To counteract this issue with limited samples only food, researchers generally use additional negative samples from general object recognition dataset, COCO is one of the most frequently used examples of this. COCO is an extremely large scale benchmark dataset containing over 330,000 images of greater than 1.5 million total object instances and spanning across 80 different object classes. Although Food101 only has images of foods, COCO consists of many varied non food objects furniture electronics vehicles household items animals environmental scenes.

Using COCO as a source of negative training samples provides a sub-

stantial amount of additional training to help make food recognition systems are more practical and robust. Since the classifier is exposed to visually diverse non food images, it learns to differentiate between food and non food content rather than assuming that all images will contain some type of food product. In turn, this establishes strong decision boundaries between food and unrelated objects to minimize the risk associated with false positives when deployed.

Moreover, in addition to simple negative labelling, the complex natural scenes in COCO will also provide another benefit, i.e. helping improve the robustness of a trained model against real world visual complexity due to the often cluttered backgrounds, the many different items present within a single image i.e. multiple objects, and the difficult environmental conditions shown in the COCO imagery.

This is especially important for food recognition systems where the dining environment may have plates, utensils, tablerunners, drinks, packaging, and other visual distractions.

Overall, using Food101 in conjunction with COCO will produce a more realistic and application oriented training environment. Food101 provides structured positive classes of food to aid detailed recognition of food categories, while COCO provides diverse nonfood samples to help increase rejection capabilities and to improve robustness to varying environmental conditions. In summary, this combination of datasets will provide systems that can not only identify certain food categories, but can also determine whether or not there are any foods present in the image.

Yet, challenges associated with merging together the Food101 and COCO datasets must be considered. Because of their differing methods of collection, presentation e.g., layout, and the types of objects represented in the images, there is a danger of domain shift occurring between positive

and negative examples. For instance, Food101 consists primarily of images where the main subject of interest is food; whereas, COCO consists of multiple types of images that include food, but also contain other types of elements including many different arrangements of objects in relation to one another thus creating diverse possible contexts accompanying images for any one image in the collection. Models may learn to use only superficial dataset specific features when distinguishing between food vs. non food, unless there is a concerted effort to prevent this from happening during training by balancing the distribution of positive and negative examples through sampling, augmenting, and preprocessing techniques [9].

In summary, benchmark datasets provide a foundational platform for research in recognizing food, and thus the ways they are constructed will have a direct impact on the resultant recognition model's ability to generalize. Food101 is regarded as the de facto benchmark for classifying food images because of the number of different classes, the variety of unique images presented for each class, and the realism of the images rendered. Conversely, COCO serves as the second major reference point because of the critical context and support for non food related objects that it provides. Creating an integrated collection of both datasets results in an effective framework for building practical food recognition systems that will be applicable in a real world application.

This project uses the Food101 dataset to train a supervised system to classify food into various categories. However, in addition to this dataset, we include "selected negative samples" from the COCO dataset to help improve the food vs nonfood classification and also to strengthen background discrimination for better performance. This means that our hybrid dataset trained system will perform well in both achieving high benchmarks accuracy as well as having a reliable practical use for assisting with

food analysis applications.

2.3 Saliency Model

As deep learning models improved, researchers started realizing that simply having a larger model wasn't enough to get the best results in complicated visual tasks. Convolutional neural networks CNNs are a great way to get hierarchical features of an image, but until now have equally considered every spatial and feature channel of the image being processed. In the real world as far as recognizing images is concerned, not all areas of the image being processed contribute equally to the ultimate prediction. When people look at a scene, they instinctively direct their visual attention to the most relevant parts of that scene and ignore the irrelevant background. Because of this underlying biological characteristic, attention algorithms and saliency modeling were developed to help direct a neural network to the most relevant parts of an image that it is working with.

The significance of attention is especially clear in food recognition tasks. Food images often have a lot of other things around them. For example, plates, forks, spoons, napkins, tables, drinks, wrappers or other decorations. Sometimes these other items take up quite a bit of space in the image and can create noise when trying to classify an image into its correct category. If you take an image and use deep learning algorithms equally across the entire image then deep learning will likely cause a correlation between the background and the food category of an image and will therefore cause a lack of generalization and more sensitivity to environmental variation. Attention mechanisms will allow you to find visual features that help identify things while ignoring all the other items that don't help identify them.

The Convolutional Block Attention Module CBAM, a lightweight but powerful add on to convolutional neural networks, is one of the key attention mechanisms that can be used to improve feature refinement [10].CBAM uses two sequentially applied attention stages channel and spatial. The channel attention stage determines which of the feature maps contain the highest information by calculating channel wise importance weights, enabling the network to place more importance on the discriminative semantic features while suppressing the rest. The spatial attention stage identifies the spatial regions of interest within the feature map, thereby enabling the network to focus on important parts of images like food [11]. The channel attention mechanism can be formulated as shown in Equation 2.3.The spatial attention mechanism is expressed as shown in Equation 2.4.

$$F' = M_c(F) \otimes F \tag{2.3}$$

where:

- F is input feature map
- $M_c(F)$ is channel attention
- $\otimes$ is element wise multiplication

$$F'' = M_s(F') \otimes F' \tag{2.4}$$

where:

- F' is refined feature
- $M_s(F')$ is spatial attention
- F'' is final output

By using a combination of channel and spatial attention, there is an increase in quality in the representational data because the attention allows the network to work with only relevant patterns and reduces interference from images that would not be considered as containing usable visual information. In very particular instances, such as coarse grained food classification, the increase in feature selectivity from channel and spatial attention will provide considerable benefits for distinguishing between foods with similar visual appearance based solely on texture or ingredient distinctions. Research has demonstrated that the incorporation of CBAM into different forms of convolutional backbones will lead to an improvement in classification results for a wide array of visual classification tasks with very little to no additional computational cost.

Attention methods focus on saliency models from the standpoint of feature maps within neural networks versus preprocessing an image in preparation for classification. The purpose of saliency models is to discover the parts of an image that will receive the most amount of attention by a human observer as compared to all other regions of the image. Therefore, instead of learning where attention should go only inside the classifier, saliency models try to create attentional maps for use during classification.

Graph Based Visual Saliency GBVS, designed as a computational model of how humans focus their visual attention, is an early and influential method for determining saliency in images [12].Using local and global contrast relationships between regions of an image, GBVS creates saliency maps that indicate areas that can be visually separated from their backgrounds. Saliency is then assigned to a region based upon the amount of colour, intensity, orientation, or texture contrast the region has as compared to its surrounding regions – the more contrast, the higher the saliency score given to the region; thus, saliency maps will emphasise those regions

of an image where there are major visual objects.

Saliency maps generated using GBVS allow for the elimination of objects in an image that do not contribute to the identification of food, thereby allowing for the identification of edible objects that stand out as such from other objects i.e. dishes, utensils, table after eliminating those that cannot be used to identify edible objects plates, cutlery or the table surface from the background. By doing so, GBVS performs a preprocessing step that directly impacts the robustness of the classifier by reducing the impact of the background and non food objects. However, since GBVS uses primarily low level contrast features for generating saliency, when food and non food objects share similarities in terms of colour or texture, GBVS will not create saliency maps that accurately identify food objects within an image due to this reduction in the availability of contrasting visual information.

Saliency methods that focused on local contrast have evolved into more sophisticated region based saliency methods, such as global contrast saliency models that compare all segments of an image with one another to identify salient regions. They give more weight to segments that are significantly different from the overall visual context of the image the entire image than to segments that are more similar. Global contrast saliency models demonstrate higher accuracy in capturing the importance of a segmented food item in an image as opposed to merely identifying it as an object because they account for both local and global contrast. If the food item does not exhibit strong local contrast with surrounding objects, global contrast saliency models will still indicate it as a salient food item.

As deep learning developed, there was a move from using only local contrast to include supervised segmentation based approaches to saliency modeling, which allowed for the development of semantically meaningful, more

accurate region masks. UNet is one of the most widely used architectures for supervised segmentation and saliency modeling, although it was originally designed for biomedical image segmentation. UNet has been adopted for general type segmentation and saliency modeling. UNet's unique encoder decoder architecture with skip connections allows it to maintain the spatial detail in a segmented food object while also creating a high level semantic representation. Therefore, UNet can produce pixel accurate region masks that correctly delineate the boundaries of a segmented food object.

When food is recognized by UNet, saliency extraction allows for very accurate localization of the food areas so that classifiers can work off of segmented food only rather than raw image. This is helpful in ensuring that the models are not influenced by cluttered environments or spurious correlations when classifying food. Segmentation also provides more interpretability as it explicitly shows which parts of an image contributed to the recognition process.

One of the other advanced segmentation frameworks that could provide better accuracy is DeepLabV3+. DeepLabV3+ uses an encoder decoder methodology combined with atrous spatial pyramid pooling to capture multiscale context i.e., capture information on multiple levels of detail [13].The use of atrous convolution expands the receptive field of the network without an increase of the computation cost, thereby allowing for better object context modeling and for better boundary detail. DeepLabV3+ has been able to develop high quality semantic segmentation masks while maintaining good boundaries, which makes this framework ideal for finding irregularly shaped foods in complex environments.

Incorporating DeepLabV3+ into food recognition pipelines allows for increased accuracy of segmentation and improved performance of downstream classification due to providing models with cleaner and more fo-

cused visual inputs. This is especially helpful for food images where object boundaries could be indistinct for example, due to sauces or overlapping ingredients, or the way food items are arranged on a plate irregular plating.

Combining attention mechanisms with saliency modeling results in two approaches that complement each other to improve recognition performance. Attention modules such as CBAM help refine the internal features produced by a neural network, thus allowing the model to place varying degrees of emphasis over different types of semantic and spatial patterns, which helps enhance the probability of success in learning. Saliency models assist with preprocessing and input conditioning by isolating areas regions within an image identified as being relevant prior to classification. By using these two techniques together, both focus can be enhanced at the input and feature level and result in creating a recognition pipeline that is more robust and better able to create a distinct output supplying more confidence in producing an accurate output.

This project focuses on utilizing attention and saliency mechanisms to provide reliable food recognition performance under real world conditions. The use of CBAM provides for enhanced feature selection in classification models, while various saliency and segmentation based preprocessing methodologies GBVS, UNet, and DeepLabV3+ are used to isolate food from its surroundings through various means. Attention and saliency mechanisms guide the system to the areas of the image that contain the most information, thus reducing background interference and improving classification accuracy in unconstrained environments.

2.4 Object Detection and YOLO Framework

Models for image classification based on their traditional methods assign a single label to an image, assuming that there is a single object taking up the most space in the image being classified. This assumption is mostly valid when creating benchmark datasets and conducting experiments within controlled laboratory environments as most images will have a single object in view. However, this model does not adequately account for real world dining experiences, where food is generally served together as part of a meal on the same plate or table. Real life examples may include ordering rice, curry, salad and bread all within one frame or multiple servings of drinks or desserts also being in frame at the same time. In these instances, traditional models for the classification network will not work because they can only return a single dominant object for the entire image and they cannot provide any information about where the individual food items are located or how big they are relative to each other in the same image. Thus, the inability of today's classification models to identify individual food items will remain a significant hurdle to successfully deploying digital food recognition systems in the real world, which will require employing an object detection based framework [14].

Instead of just identifying objects i.e., types of food in an image like with classification; object detection identifies in addition to the object's existence where in the image that object is located. Object detection models provide multiple predictions of image contents by generating several bounding box type annotations, each with a class identified object type and a confidence score. These capabilities now allow food recognition systems to achieve scene understanding; this means being able to identify many different types of foods within one image.The performance of object

detection is evaluated using Intersection over Union as shown in Equation 2.5.

$$IoU = \frac{Area_{Overlap}}{Area_{Union}} \tag{2.5}$$

where:

- $Area_{Overlap}$ is intersection area
- $Area_{Union}$ is combined area

YOLO You Only Look Once is one of the most significant frameworks for object detection among many that have been created over the last ten years due to its speed and accuracy. After being developed by Redmon and other authors, the first YOLO architecture changed the way people think about object detection by converting object detection into a single stage regression problem instead of a multi stage proposal and classification system. Prior to YOLO, object detection systems like RCNN and Fast RCNN used region proposal methods to find object areas in an image and then to classify each area. While these methods were accurate, they were too slow for use in real time applications because of the resources required to compute region proposals.

The difference is that YOLO performs a single forward pass through the entire image by using the entire image to make predictions about object bounding boxes and class probabilities as a function of the grid into which the image has been divided. As opposed to making predictions based on region proposals separately from each other, YOLO divides the image into a grid and predicts the object bounding boxes and class probabilities for each grid cell at the same time. Compared to utilizing region proposals separately from each other, YOLO greatly improves the speed at which inference can occur for real time object detection while achieving comparable accuracy when compared to standard techniques for object detection.

There have been many changes made to future versions on YOLO have been made to increase accuracy of locating items within an image, detect small sized objects, and improve stability of training; resulting in an overall improvement in the YOLO family of detection models becoming one of the most utilized detection frameworks in both the academic and commercial sectors where speed is essential such as autonomous vehicles, video based surveillance, robotics automated systems, and mobile computer vision.

The newest improvement of YOLO which is relevant to this study is called YOLOv8. It was created by Ultralytics as an updated version of the YOLO framework for object detection and incorporates several architectural enhancements, updates to training techniques, and optimization to the inference pipelines. YOLOv8 uses anchor free detection heads, improved feature pyramid fusion, enhanced loss formulas and streamlined architectural design that should provide much higher performance levels for a broader range of object detection tasks than prior versions.

Another key component of YOLOv8 in the context of food item recognition is the capacity to detect multiple food items in scenes cluttered with multiple plates of food. Unlike classification models that return only one label, YOLOv8 separates and independently labels each item, generating unique confidence scores that provides a more accurate analysis of multiple component meals while providing detailed nutritional information about each identified plate of food.

Another major advantage that YOLOv8 provides is its ability to operate with low latency inference for real time assistive systems to assure effective usage for users by means of reliable low latency feedback via sound or sight once they obtain the necessary information from their assistive devices such as hearing and seeing, but in or around the same time that they are requesting this assistance from their device.

Therefore, because of YOLOv8's innovative design and increased efficiency over previous versions of YOLO, it will provide substantial improvements to assistive applications such as mobile based food recognition systems.

Training object detection models for food recognition is challenging for a number of reasons. One reason is that food is often irregularly shaped, has soft edges, often overlaps with other items, and there is a lot of variation when looking at multiple foods within the same category. Unlike objects like cars or pedestrians which generally have well defined geometric features, foods have very loose shape characteristics, which makes exact localization much more difficult. Also, on plates, foods often occlude one another in various ways which can also complicate detection scenarios.

Mosaic augmentation allows for drastic increases in the density of objects seen during training by artificially increasing the number of objects available to each training image. This results in a larger and more diverse set of knowledge for the detector about how to recognize objects found in more complex scenes. This is critical for building a robust and generalizable food detection model.

In addition, mosaic augmentation will also help improve the model's ability to learn contextually independent features by presenting items in diverse settings and configurations around them. When used in conjunction with typical augmentation methods, these benefits will help build a robust and accurate food detection model.

Another obstacle with food finding items is differentiating food items from non food backgrounds, i.e. plates, bowls or cups, utensils, drinks, and other decor items. If a food detection system has trained only on food samples, with minimal to no variance provided by training, it could result in unwanted object localization.

Negative background samples will be trained on and non food background variations will be used to aid in the identification of objects and decrease the amount of false positive findings during training.

The use of an object detecting systems such as YOLOv8 represents a massive leap forward in the development of helper food analysis systems; instead of only being able to recognize food images separately from their backgrounds, object detection systems offer opportunities for deploying in the real world with situations that are complex as far as food and non food objects in the same visual experience; this makes the system much more useful will eventually increase the potential of human assistance through an accurate means of assisting humans with their food intake.

In this project, the YOLOv8 framework has been integrated as the official Food Detection module, which is used to find and identify prepared meal items that have been identified from one image frame. The food detector has relied on food specific labels for training and has been validated against multiple different augmentation techniques to improve its robustness when it is deployed in the real world conditions and or dining environments. Functioning as an object detector as well as implementing classification, the system will allow for identification of prepared meals to occur; thus, help prepare more food for a specific meal [15].

2.5 Data Augmentation Strategies

Deep learning models are less effective in testing if they were trained with a small number of images that are not diverse or not representative of reallife conditions. Even if a deep learning model's architecture is sophisticated and has been tested for all possible scenarios e.g., on the lab, in practice e.g., in the field, a failure may occur because of a lack of variation

or representativeness in the data used to train the model. The same is true for food recognition—food images taken in the real world can differ substantially on lighting, angle, scale, occlusion, plating arrangement and background. Therefore, if a neural network is trained with only tidy lab images of food items, the accuracy of that neural network may suffer when placed in nontidy real world environments; conversely, if a neural network is trained using only diverse representations of food items, the accuracy of that neural network may improve significantly.

Data augmentation is a technique that has gained popularity in recent times to help overcome this issue of lack of diverse representations; for example, it is used to apply controlled variations to the images included in training datasets to artificially create a more diverse image dataset without collecting new images e.g., which are usually either very costly, time consuming, or inconvenient. Techniques used for data augmentation can include modifying an original image by changing the size, contrast, brightness, etc. as well as creating a variety of images from just one image. As such, the use of data augmentation techniques can allow neural networks to learn generalized representations of food items, making them less sensitive to differences caused by environmental conditions [16].

Augmentation is critical for food recognition systems due to the amount of variability in how food looks. The way food is presented can cause a large number of visual differences, depending on the cooking methods used to create the meal, the number of ingredients used to create that dish, how it has been decorated garnished, angle of the picture taken, or any number of other factors around the dish. If a classifier simply memorizes a narrow set of visual characteristics from training images, it will often not be able to classify a dish after its presentation changes. Using augmentation allows the model to see multiple variations of each image during training, which helps

the model learn stronger semantic features than would have resulted from just learning those images based on how they were originally presented [16].

Geometry based transformations such as rotation, translation, scaling, flipping and cropping are among the most commonly used augmentation strategies in deep learning. Geometric transformations simulate the natural variation that occurs with the location of the camera and the orientation of the object in relation to the camera. In food recognition, users may take pictures of meals at different angles, from different distances or different positions of their hands. Each of these variations will produce significant perspectives. When applying geometric augmentations at training time, the model will learn features that are orientation invariant and scale invariant; this will increase the robustness of the model relative to how a picture of food will be captured in the real world.

Rotation augmentation works great for food images. Their semantic identity has little or no change as an image's rotation changes. This is different from recognizing text or identifying faces. If models trained on food images are shown rotated food images to train on, their result classes the labels identifying the classes of foods will not change, just the images will be transformed from their representations in the training set to the rotated versions of those images. In addition, this also applies to scaling and cropping; these techniques develop the knowledge of how to look at an object and replicate it from different distances and framing orientations.

Another very important category of augmentation techniques is photometric transformation. Photometric transformations are those that change the appearance of an image without affecting the geometry. This includes brightness, contrast, saturation, hue, and sharpness variations among food images captured in real life environments—meaning there is not a uniform illumination of food images due to various factors such as indoor lighting,

shadows, sunlight, reflective surfaces, or low light environments. Therefore, without applying photometric transformations to food images, the model will generally become too sensitive to visual cues related to light and perform poorly on images of food captured at different light levels from different environments.

To provide a solution to this problem, photometric transformations that are typically utilized during training are photometric color jittering and photometric brightness and contrast variations. Photometric transformations such as these yield random perturbations of all visual cues related to brightness of the image and thus lead to models that rely less on the absolute intensity of the color of the food objects and instead rely on stronger structural and textural features when classifying them. This ultimately results in better generalization of learned representations in different environments and greater reliability during deployment of the model.

In addition to manual augmentation techniques, there has been a growing trend toward using automated augmentation techniques to increase the overall effectiveness of augmentations. RandAugment is one such automated augmentation framework that provides a simplified and effective means of generating automated augmentation techniques [17].Traditional augmentation workflows require the user to manually select and tune the type, probability and magnitude of each of the augmentation operations; this is typically a time consuming and expensive process. RandAugment randomises the selection of augmentation transformation from a specified pool of augmentation transformations and applies transformations with standardised magnitude controls.

Because of the simplicity of the approach used by RandAugment, RandAugment has been found to perform extremely well empirically. By introducing randomised transformation diversity during the training phase

of machine learning models, there are a large number of synthetic image samples available for the model to learn from which allows the model to become more robust and therefore reduces the chance of the model overfitting. Additionally, RandAugment is particularly advantageous when applied to food recognition tasks because it enables the user to simultaneously recreate multiple real world distortions without the need for large amounts of time or effort spent in designing manual policy.

In the case of object detection frameworks such as YOLOv8, augmentation techniques extend beyond the application of single image transformations, to the use of compositonal augmentation techniques. One of the more popular among these is known as mosaic augmentation. Mosaic augmentation involves taking four independent images and combining them to create a single composite training sample for the object detector; this allows the object detector to view more than one object located within different scales and locations from one another, therefore improving the object detector's ability to recognise objects in very crowded or multi object environments.

The use of mosaic augmentation in applications that detect food items is valuable to the application because it is common for food items to be arranged together on a single plate in an image. Mosaic augmentation provides the food detector with examples of a synthetic multi food, multi dish environment during training. This allows the detector to learn to detect the contextual independence of multiple dishes, thus improving its ability to locate dishes in a true restaurant environment.

Another augmentation technique that can help the food detection system is called noise injection. In this technique, random visual disturbances e.g., blurring, Gaussian noise, compression artifacts, and occlusion patches are injected into training images to help create training images that will

simulate degradation of real world images e.g., camera motion, out of focus, poor quality camera sensor, and partial occlusions. Because users using mobile devices may capture images that do not meet optimal image quality standards, by training the food detection system with noisy and or degraded images, the system will have greater robustness to the imperfection of when the food detection system is deployed in the real world.

The cumulative impact of augmentations is far greater than only improving benchmark accuracy. In increasing the amount of effective dataset diversity of the food detection system, the food detection system will be subject to less overfitting and will therefore generalize much better than a standard non augmented food detection system. In this regard, a food detection system that has gone through extensive augmentation will be less dependent on dataset specific artefacts and will be able to extract semantically relevant visual representations, thus constituting a reliable system for food recognition where assistive technology will be used and where reliability of the system operating in uncontrolled environments is required.

In order to apply an augmentation method correctly there must also be some consideration of how much industry standard augmentation or an extreme amount of it will cause confusion and therefore affect the model's learning. Augmenting with extreme or an unreasonable amount of colour shift, for example, may change the food to such a degree that the food appears to no longer match its original identity. Similarly, using aggressive cropping techniques will remove important feature information from the food item. Therefore, augmentation strategy development requires creating a balance between the enhancement of diversity through augmentation and the preservation of semantic meaning.

In this research project, the augmentation of both geometric and pho-

tometric techniques and advanced augmentation strategies will be used in the training of the system. These include using techniques such as rotating, scaling, brightness adjustment, implementing a ColourJitter, RandAugment and mosaic augmentations in the training pipeline to simulate a realistic environment when deployed. The augmentation strategies described in this paper will allow for sufficient robust and generalisable food recognition systems to operate effectively across different real world conditions.

2.6 Models Integration

Although significant advancements have been made in developing the fields of Computer Vision and Deep Learning particularly within Object Recognition and Food Analysis most of the advancements have primarily been focused on technical metrics such as improved Classification Accuracy, increased Detection Precision and Computational Efficiency. The shift from purely technical metric improvements to converting those technologies into practical Assistive Technologies systems capable of providing functional assistance to a person with a disability is relatively new; however, in many cases, Assistive Technologies designed for individuals with Visual Impairments are not sufficiently usable or accessible unless they are designed with an emphasis on Accessibility, Usability and Independence; therefore, the most accurate recognition model will not be able to assist an individual with a Visual Impairment if it is not integrated within an accessible interface.

Assistive Technology is defined as any device, system, or software application utilized to augment improve the functional abilities of individuals with disabilities and promote Independent Living by providing access to

everyday activities. Traditional Assistive Technologies for individuals with visual impairments include basic Mobility, Accessing Printed Material via OCR, Braille, & Accessing Digital Content screen readers, etc. While traditional forms of Assistive Technology have greatly improved individuals' access to traditional forms of information e.g., text and the ability to navigate independently; Access to Food is still a relatively unexplored area given its critical importance to day to day life for everyone including those with disabilities.

Identifying food items independently can offer numerous problems for people living with vision impairments. Food is a constantly changing and very visual object when compared to printed text or digital user interfaces; thus, food cannot be interpreted by utilizing typical tools for accessibility. People with vision impairments rely on senses such as smell, texture, and assistance from others when they identify food items, but relying on those senses can be limited and less effective than traditional methods. Using the sense of smell alone to identify food does not always produce consistent results because of the different smells created by similar looking food items. The same applies to using the sense of touch to identify food items; sometimes, this form of identification can be uncomfortable when out in public. Being reliant upon a person to confirm a food item can negatively affect an individual's independence level. The gap created by these limitations will give developers of artificial intelligence AI systems a great opportunity to create a bridge between how we perceive food visually and how to quantify access to it through other sources [18].

Current research in the field of Interactive Technology includes ensuring that any assistive technology system created, must have usability, clarity, and cognitive simplicity as the priority over the technical design or functionality. Assistive technology that is technically complex and has

the requirement of visual verification and or does not provide for simple methods of interaction will not be accessible regardless of the effectiveness of the technology. Therefore, adherence to accessibility principles must be built in as part of the design process and not added at the end as an afterthought to enhance a product.

Auditory feedback is an important accessible element of visual assistance technology. Because users who are blind rely on the Web server for visual output but cannot see it, the response they receive from the server must be converted to speech in a structured, predictable way so that it can be understood by the user. TTS text to speech works by converting the results from a computer's recognition system into a synthetic voice so that a user can receive information, such as the food classification, nutritional content, and system prompts, from auditory output instead of visual output. At this point, we are changing the way raw computational evidence or predictions are produced into useful, accessible, actionable information [19].

The way in which auditory output is processed and presented to users directly affects its effectiveness. Research on accessibility indicates that auditory feedback must be structured in a logical sequence so as not to overwhelm the user with too much information. For example, in a food recognition application, if the name of the food is stated first, then the ingredient information next, followed by the calorie information, a natural and logical auditory hierarchy is created. The structured communication will allow the user to understand and process the information better because it corresponds to a mental model used to create cognitive load during an interaction.

Aside from the audible tone of the output the auditory aspect of output the ease of using the interface is an extremely important factor in the

implementation of accessibility. In the design of traditional mobile applications, there are generally complex graphical user interfaces, small touch targets, visual menus and multi layered navigation systems which make it very difficult for users who cannot see to independently navigate the application [20].To design a mobile application that is accessible, there needs to be a simplification in the interaction workflow, by eliminating unnecessary interface elements and providing a user the fastest possible pathway through direct action.

In terms of mobile based food recognition systems, this means reducing how many steps are needed for the user to perform the recognition process. The ideal system will require the user to take a picture of the food item and be provided with a recognition result with very limited navigation and manual configuration. Using large accessible buttons, voice prompts, gesture based interaction and a streamlined sequence of work will all contribute to a more inclusive experience.

Responsiveness in real time is another important consideration. Supportive usability can be seriously undermined due to delayed recognition results and can lead to uncertainty while interacting with an assistive technology device. Therefore, assistive technology devices must provide immediate feedback to instill user confidence and use it practically. For food recognition applications, users expect to receive near instantaneous results after taking a picture of their food. If there are long delays in processing, users may not be able to continue their natural dining experience and will lose trust in the system. Therefore, both model efficiency and inference speed are factors of accessibility in addition to being technical performance metrics [21].

Error handling is also very important when developing support systems. No recognition system is 100% accurate and applications aimed at

providing accessibility should consider how best to handle uncertain situations. Instead of providing users with possible misleading and confident predictions in uncertain situations, it would be more effective to communicate uncertainty in a transparent manner. For example, suppose a food identification application recognizes an object as being food but does not have a certain level of confidence or below the established threshold level of confidence to provide a valid prediction. In that case, the application could inform the user that it was unable to identify the food item with a sufficient level of certainty and prompt them to take an image. By doing this, it will increase the user's confidence in the assistive technology system, and it will limit the possibility of providing the user with erroneous information.

The principle of context aware accessibility is an additional principle that is currently developing in assistive design for AI. It defines an AI system's ability to adjust the output from the system based on the user or environmental context. An example of this would be if a food recognition system provided shorter feedback for a fast recognition scenario and more detailed information regarding the nutritional breakdown of that food upon request of the user. The layered accessibility design of this system allows the user to receive the appropriate amount of information at the appropriate time while avoiding any unnecessary cognitive overload.

Ultimately, the combination of AI and accessibility principles allows for food recognition to become an assistive system that provides social value to the user and improves their ability to independently manage their diet, improve their awareness of their nutrition, and reduce their reliance upon others for assistance. Rather than simply being an identification system, the food recognition system has become an intelligent companion that allows visually impaired users to become more knowledgeable about their

dietary choices.

This Project emphasizes the integration of accessibility into the design of the AI Augmented Food Analyzer as a core design objective rather than a secondary feature. Real Time text to speech output, simplified mobile interface design, low latency processing, and structured auditory feedback mechanisms are all designed to make the recognition capabilities of the food analyzer usable, easy to understand, and meaningful to users who are blind or visually impaired. Inclusion into the project through the integration of assistive technologies will also aid in the development of an inclusive and Human Centered assistive technology system that goes beyond just food recognition research.

2.7 Summary

The body of literature reviewed in this chapter affirms that developing an accurate and effective food recognition system requires more than simply employing a particular deep learning model. Food recognition is a multidisciplinary problem that exists at the crossroads of computer vision, fine grained image classification, object detection, saliency modeling, data augmentation, and accessible system design. Each individual area of research discussed in this chapter contributes an essential aspect to solving the larger problem of producing reliable, assistive food analysis tools.

The review of deep learning architectures demonstrates how rapidly image classification methods have evolved from traditional handcrafted feature extraction methods to complex convolutional neural networks. In each case, the different deep learning architectures that were reviewed illustrate that no single architecture performs satisfactorily for all application scenarios. Each architecture makes trade offs between the quality of the

results produced accuracy, efficiency of computation efficiency, and degree of difficulty to deploy feasibility. The validation of multiple pretrained architectures in the proposed system is supported by this reasoning; thus, assisting in the determination of the best deep learning architecture for providing assistive food recognition.

Benchmark datasets are evaluated to result in not only dataset quality but also diversity being equally important to the architectural design of all models. Food101 is a tremendous benchmark for food related classification because it has a large scale, has a large number of categories, and has a large diversity of sources images that realistically represent what we call food. The COCO dataset complements Food101 in providing many nonfood contextual representations to help improve the model's ability to discriminate the food from the context i.e. remove the probability of seeing other objects in the same location as the food and therefore reduce the need for false positive predictions. As a result, both of these datasets provide valuable tools to build models that will have the capability to perform in 'real world' unconstrained image environments.

The investigation of using attention mechanisms and saliency models has shown how visual attention improves the reliability of recognition among models. For example, the use of an attention module such as CBAM helps in refining features in deep neural networks, while using saliency and segmentation techniques like GBVS, UNet and DeepLabV3+ help in focusing on relevant regions of the food and in removing any distractions due to surrounding environments. These strategies are critical to address what has been one of the most significant problems to date in the area of food recognition i.e. to ensure that the model's recognition of the food is based on the food itself and not on anything else in the model's background.

Research in the field of object detection indicates that practical food

analysis systems must go beyond labeling food with only one classification to include recognition of multiple items. Real time detection and location of object recognition capabilities for food analysis will greatly enhance the way we perform food analysis in a natural, social dining environment. An example of a good framework for real time detection and location of multiple foods is through the use of YOLO regardless of which YOLO version we are using for example, YOLOv8.

Research on data augmentation continues to show robust return on deployment performance depends on diverse synthetic visual variations available during training. Techniques such as geometric transformations, photometric distortions, RandAugment and mosaic augmentations are examples of how to improve the generalizability of a model and narrow the gap between data sets from published sources and deployed performance in everyday life.

Based on the literature regarding assistive technologies, simply having accurate technical recognition does not provide enough value to visually impaired users. A successful integration of assistive technology requires accessibility features such as auditory feedback, simplified interfaces, low latency, and an inclusive interactive design to convert a technical recognition model into a valuable assistive technology that will increase the independence of users when they perform tasks on a daily basic.

The reviewed body of research has established a solid theoretical and practical base for the AI Powered Food Analyzer. The existing body of literature indicates that AI Powered Food Analyzer needs to be built using a unified intelligent pipeline composed of deep classification architectures with detection frameworks; saliency enhancement; augmentation strategies; and principles of accessibility. This collection of advancements has provided insight as to why many current food identification technologies

focus on the various technology components and not all of the interconnected advancements within a single holistic, assistive technology framework designed for the visually impaired end user.

In light of the identified areas of future research and limitations identified, the proposed project will develop a unified, mobile based, assistive food analysis technology using multiple classification backbones utilizing pretrained models; saliency guided preprocessing; YOLOv8 food recognition; access to a nutritional database; and an auditory access method to provide auditory feedback as a whole, practical system. The next chapter presents the requirements and specifications that will be used to guide the design and development of the proposed solution.

Chapter 3

Requirement Specifications

To develop an AI Augmented Food Analyzer, a comprehensive approach must be taken when evaluating the needs of the system, the user requirements, and the limitations of its implementation. Requirement specification goes beyond just being a documentation task but creates the basis on which the actual system will be built. For projects designed to help people who are blind, requirements must also incorporate other factors including accessibility, reliability, scalability, and adaptation to real world conditions as well as providing accurate calculations. This system, unlike traditional food recognition applications designed for individuals with vision, is based around promoting independence and non visual interaction. As a result, the requirement analysis that follows in this chapter focuses on evaluating existing systems, defining the architecture of the proposed solution, listing functional and non functional requirements, as well as determining feasibility and constraints.

3.1 Existing System Analysis

Determining what existing systems are capable of doing is one basic component and the starting point for developing requirements specifications. In creating the design of an intelligent system, you must first have a good understanding of what current systems can and cannot do before establishing what to develop. A new system can only be relevant if it addresses real world deficiencies of existing systems' capabilities rather than simply duplicating current capabilities. Therefore, the analysis of existing systems is both a technical and strategic task in the software engineering process because it helps to identify quality and performance gaps usability issues, performance issues and user needs before designing a new solution.

According to research, the most popular type of existing dietary sup-

port system are mobile nutrition and calorie tracking apps such as MyFitnessPal that provide users the ability to enter food items they consume manually, as well as record portion sizes, through a searchable database of nutritional information. Research indicates that these types of applications can affect user dietary behaviors by either increasing users' awareness of their nutritional intake or by promoting more healthful dietary choices through continuous monitoring of their eating behavior [22].Digital diet help technologies are effective ways to give people the ability to understand nutrition. Nevertheless, the manual entry approach to some digital diet help applications creates a big usability problem. Users have to do the same task repeatedly; finding food, estimating portions and logging in food creates an interaction burden that gets boring over time leading to low continued use. For users looking for a quick and easy way to use these types of applications, the repetitive nature of the workflow creates issues that make using the application convenient.

In corporate applications, there are even more significant barriers to accessibility with regard to the manual entry nature of nutrition applications. Current digital diet help applications presume a user can see how to access the menu, read the food label, find the food database and make a selection through graphic type interfaces. As a result, these applications are not usable by users who are blind and might find the traditional touch screen interfaces cumbersome, slow and or impossible to complete without substantial support from a third party. Therefore, digital applications currently function well for sighted users who have the capability to manually enter food into an application, but do not provide a user with a truly autonomous experience for dietary analysis via non visual interactions.

An additional common food identification approach is the barcode scanning system that has been incorporated into various commercial fitness

food tracking applications and grocery shopping applications. Barcode scanning systems are able to recognize a product by scanning the machine readable label on the outside of the food and providing access to the associated nutrition information that is stored in a database and linked to the scanned product. This scanning approach uses deterministic matching of identifiers to identify food, rather than using image recognition or inference [23].Because of this, barcode scanning systems provide a highly accurate and efficient recognition method for prepackaged, commercially produced food items. Furthermore, barcode scanning is very computationally lightweight, highly dependable and well suited for the types of structured retail environments typically found in supermarkets.

Their convenience for both purchased and ready to eat packaged items means that the limitations of barcode based systems are amplified when applied to less restricted situations for food recognition in a more general way. Simply put, barcode based systems can be used to identify items with a visible, scan able barcode but cannot be used to identify freshly prepared meals at a restaurant, home, and mixed combined plates and unpackaged foods i.e., produce or prepared foods. Because of this restriction to products with a barcode, barcode based systems cannot be used to help with many day to day diet related situations i.e., diet related food consumption and food preparation or with helping people connect with and use content in a real world, practical way to help them meet their nutritional and dietary goals e.g., health, frozen pizza, etc. Moreover, scanning a barcode requires users to find, vie, and position the barcode correctly with respect to the camera field of view; therefore, while barcode systems allow for the automated lookup of items, they do not provide a solution for the more significant challenge of food recognition in an unconstrained dining environment.

AI powered diet programs with AI integrated Machines Learning personalized recommendation systems are emerging rapidly recent years. Unlike calorie counting applications calorie counting applications that provide static recommendations based solely on a person's calories consumed, this technology offers dynamic personalized planning based on user attributes for example: age, weight, fitness goals, calorie goals and food preferences [24]. The improved personalization offered in the recommendation capabilities of these types of platforms provide increased user engagement when utilizing the technology by providing contextually based in relation to the user versus standardized common or universal nutrition recommendations.

Although there is a growing number of AI systems, many of the current systems still rely on manual logging logging food consumed of food items consumed because food does not get recognized as an input through the process of AI systems functioning through machines to produce nutrition analysis and provide specific recommendations. Because of this, even though these AI systems have made substantial improvements in the quality of their recommendations, they cannot eliminate the challenging usability burden experienced by users of these AI applications' current form associated with having to perform manual meal entry. Additionally, the typical user interface design of the current AI systems tend to be focused on the average user and not consider incorporating alternative interaction methods geared towards making the systems accessible to individuals with disabilities. As a result, most current AI systems do not include any forms of automated feedback through audio voice, visual video or tactile vibrational cues to user or simplified user interfaces.

There are many computer vision based food identification systems in the research and commercial spaces from a technical point of view. Com-

puter vision and deep learning model techniques are employed for food image classification using user captured food images. However, many of the current food recognition applications that can be achievable will also have limitations in their real world applications, such as limited classification range, not being robust enough to handle messy surroundings, and being able to only classify images with a single label. Most computer vision classification systems rely on a single, large, or dominant food item occupying the whole image, whereas in reality, multiple dishes may be on one plate or in the overall picture of the table that is being served. This means that when using these types of systems that rely on one item being classified, people will not be able to use the system when they are served multi item meals.

Many existing food recognition systems exhibit a significant limitation in their ability to correctly interpret complex environmental conditions. For example, many types of dining environments often contain various visual obstruction factors such as cutlery, plates, tablecloths, shadows, packaging materials, other background items, and different types of lighting. Consequently, systems that have only been trained on clean and standardized benchmark datasets typically perform successfully under laboratory conditions but will suffer considerably when deployed in real world scenarios. In evaluating the performance of a technical system, human factors and usability principles dictate that we examine not just the laboratory performance of the system, but the effectiveness, efficiency, and reliability of the system as appropriate for the actual context in which it will ultimately be used [25]. Therefore, a recognition model that can identify food accurately in a formal testing environment but is not successful in identifying food in informal dining situations cannot be regarded as a useful assistive technology for deployments made for the purpose of assistance.

There are numerous assistive technologies available today for people who have visual impairments, but unfortunately there are significant limitations in how they can be used for dietary support. Although screen readers, voice assistants, and object recognition tools all enhance general accessibility, they are not typically made for fine grained recognition of food items or analysis of the caloric and nutritional content of foods. For example, general purpose object recognition will allow a person to know that there is food on their plate, but not necessarily what type of food is present or how many calories are in each of those items. This limitation demonstrates that while accessibility may provide an adequate level of support for many types of tasks, there still needs to be specialized knowledge oriented toward the specific use case of dietary assistance in order to be considered to provide adequate support.

Taking a general perspective of system design, it is evident that all of the current solutions available today represent a patchwork solution to the overall problem of supporting someone with their dietary assistance. Stand alone nutrition applications provide nutritional databases to users, but require a significant amount of time interacting with the application by entering information regarding the foods that they consume. Barcode recognition systems provide a structured method of recognizing food through barcode scanning but only apply to prepackaged products. AI based recommendation systems will provide personalized recommendations based on the user's past consumption patterns or preferences; however, they still rely on the user to enter information regarding the foods that they consume. Image based classifiers automate the process of recognizing food through image file classification; however, many do not provide sufficient robustness, accessibility integration, or readiness for use in real world environments. Only a very limited number of the existing assistive

technologies have combined functionalities of automated visual recognition, multi food detection, accessibility oriented user interface functionality, nutritional analysis, and real time usability into a core, unified solution.

The limitations described earlier in this paper point to the need for the proposed AI Augmented Food Analyzer. This proposed system aims to overcome the limitations inherent in existing solutions by combining elements such as automatic image based food recognition, real time object detection, saliency based preprocessing, nutritional mapping, and auditory feedback that is accessible into a single mobile based assistive platform. The intended system will reduce user interaction through the automation of the process of recognizing food as compared to traditional manual entry dietary applications. In contrast to barcode based systems, which require that meals have machine readable labels, the system will analyze food visually. In comparison to traditional image classifiers, the proposed system will include multi object detection and saliency techniques to enhance robustness in real world dining situations. Finally, unlike other mainstream food recognition tools, this system is specifically designed for the visually impaired user, providing for an accessible interaction and independent usability. In conclusion, based on the current state of research about systems that analyze foods and dietary approaches, there is no solution available that represents a completely sufficient set of tools to provide visually impaired people the ability to independently analyze foods due to the above referenced limitations of the various food and dietary recognition systems. Therefore, there is evidence supporting the need and justification to create a combined Artificially Intelligent AI Food Analyzer as a potential option for filling the gap between visually impaired persons' desires for independent food analysis and the lack of existing technology that meets those needs.

3.2 System Framework

Following the analysis of existing systems and their limitations, the next step in requirement specification is to define the conceptual framework of the proposed solution. A system framework describes the structural and operational blueprint through which all individual components interact to achieve the intended functionality of the application. In software engineering, the framework serves as a bridge between requirement analysis and detailed design by translating identified user needs into an organized functional architecture. It establishes the logical relationship between data acquisition, processing modules, inference engines, databases, user interfaces, and output mechanisms, thereby providing a high level understanding of how the proposed system will operate as an integrated whole. As shown in Fig. 3.1, the system integrates mobile application, cloud server, and deep learning models.

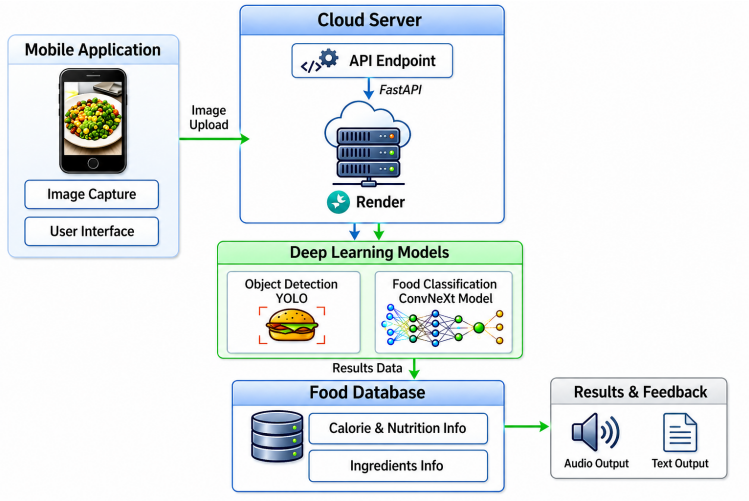


Figure 3.1: Overall system architecture of the proposed food recognition system

For the AIAugmented Food Analyzer, the proposed framework is de-

signed as a multi stage intelligent processing pipeline that transforms raw visual input into meaningful and accessible dietary information for visually impaired users. Instead of looking at food recognition as simply classifying food, the system used a structured approach to perform food recognition based on different parts of the process including image processing, intelligent food recognition, nutritional information of the food and generating user feedback based on how accessible the food is to eat. The modular design of this system means that it will keep an acceptable level of food recognition performance while being easily scalable, maintainable and flexible to support improvements in the future.

The first step in operations is to acquire the image of the food being consumed by taking an image in real time of the image of the food through the cameras' image acquisition component of the mobile application. The use of this camera component of the application will be the system's visual sensing component and will be the first component of the food recognition pipeline for food recognition. In order for this system to be used for food recognition in real world settings where it will be used on mobile devices, the image acquisition component must be able to provide the user with fast image captures of food, an easy way to use the system to capture the image of the food and have the capability to handle various camera orientations and natural light conditions. The quality and consistency of the captured image of the food will have a significant impact on the food recognition performance of the system, which makes this step a very important step for the overall reliability of the food recognition system.

The visual data from the image captured as part of deep learning inference is preprocessed and enhanced by going through its preprocessing and enhancement phases to prepare it for further processing. The common types of preprocessing applied to the image are resizing of the image, per-

forming a normalisation of the image, converting the image to the required format to be compatible with the input specifications of the trained model. In the case of food images, the image may have quite a lot of background noise due to cluttering and other distractions around the food image; as a result of this, the framework incorporates saliency guided enhancement techniques to improve the ability to accurately identify the relevant visual regions. Some of the techniques that may be used to assist in saliency extraction or the segmentation of the food image are the Graph Based Visual Saliency, UNet segmentation, or DeepLabV3+ so that the regions containing food can be isolated or enhanced, while at the same time minimising the irrelevant parts of the image such as the plates, silverware, table surface and surrounding objects. As a result, the processing of the visual input at this stage before it is classified, allows for more robust recognition and decreases the likelihood of being classified incorrectly due to background noise.

After the image has gone through preprocessing, it is then sent to the classification and detection layer, which serves as the brain of the proposed framework. For single food scenarios, the system will utilize multiple deep convolutional classification architectures ResNet101, EfficientNet, ConvNeXt, MobileNetV3Large to determine the most likely food category based on learned visual features. The various pretrained architectures used allow for comparative evaluation of the performance, efficiency, and suitability for deployment.Additionally, attention mechanisms such as the CBAM can help improve the intermediate feature representations created in the model by directing the model's attention towards the most discriminative spatial or channel wise [26].The classification probability is

computed using the Softmax function as shown in Equation 3.1.

$$P(y_i) = \frac{e^{z_i}}{\sum_j e^{z_j}} \tag{3.1}$$

where:

- $P(y_i)$ is probability of class i
- z_i is logit score
- $\sum_j$ is sum over classes

In the case of complex meal scenarios multiple food items in an arrangement, the framework goes beyond pure classification of the food through the integration of an object detection module that utilizes YOLOv8 based object detection. Using this object detection module, the framework will detect and localize the many different items of food in an image providing a unique label and bounding box for each detected food item. Therefore, the use of object detection substantially increases real world applicability of the framework since the vast majority of practical meal scenarios have two or more types of food arranged on the plate table at the same time. Without this ability for object detection recognition in the Framework, the system would be limited to simplified single food identification scenarios and, consequently, be unable to provide realistic meal structures.

After the recognition stage has completed, the predicted food labels are sent to the nutritional analysis module using the recognized food classes where the recognized categories are mapped to a nutritional database in a structured format that contains detailed metadata on food food specific. The nutritional database includes ingredient lists, approximate calories and possibly macronutrient breakdowns associated with each recognized food class. By linking the output of the visual recognition with the associated

metadata in the nutritional database, the framework produces useful dietary information for use by the users after receiving the raw predictions from the classification module due to the fact that food alone does not fulfill the overall intent of assisting in making well informed decisions on dietary choice. The user must also know what is in the food as well as the nutritional aspect of the food.

After retrieval of nutrition information from the database, the application then moves to the accessibility and feedback layer where the system generated output display display inferencing is transformed into various formats that allow for interaction by users with a vision impairment. Text to Speech synthesis is utilized to convert the result of an image recognition process into an audible output so that the user can hear the food product name, estimated calorie count, and ingredients without needing to view the screen. The way audio output is structured i.e., sequence, clarity is designed to ensure maximum understanding and usability. For example, the food item will be announced first, then the total number of calories in that item, and then the individual ingredients in that food item. By providing organized and deliberate audio output in this manner will improve the user's cognitive processing of the information and conform to best practices in disability access to assistive media.

In this project, a critical architectural feature of the design is the use of server based deployment for the separation of mobile interface logic from computationally intensive inference processing. For example, deep learning models like YOLOv3 and ResNet101 require large amounts of compute resources to process images for object recognition. Therefore, while the Android app will be the primary user facing client, the trained models will be hosted on a server side backend using a mobile client server architecture; mobile clients will send captured images securely to the inference server

where the images will be processed through an inference pipeline that has been deployed in the server and then return predictions to the mobile client application so users can see feedback. This architecture will help reduce the load on mobile hardware, increase scalability, and allow for updates to recognition models to occur on a centralized basis without having to reinstall the application.

In addition to improving overall performance, the modularity of the proposed architecture will also improve the framework's maintainability and extensibility. Processing components e.g., preprocessing, classification, detection, nutritional mapping, and feedback generation are grouped into independent but interrelated modules that can be upgraded piecemeal in the future without having to redesign the complete system. For example, if an upgraded classification model becomes available, then it can replace a previous architecture; if a larger food data set becomes available, it can be added to the existing database; and new accessibility features can be incorporated without disrupting the overall framework. The modularity of the proposed design follows the principles of modern software engineering by designing for maintainable and scalable architectures.

The proposed framework integrates computer and mobile application development, visual recognition, and dietetics into a single holistic system. Unlike existing food recognition applications that are isolated from diet analyses, the proposed system provides end to end user interaction from image capture through visual recognition, diet nutritional definition, auditory feedback, and energy planning.

The proposed framework defines an architectural model for the AI Augmented Food Recognition System AIFRS as a modular intelligent pipeline containing: 1) image acquisition; 2) image preprocessing; 3) image saliency detection, enhancement, and availability; 4) image classification; 5) object

detection; 6) nutritional mapping; and 7) user selected auditory feedback methodologies; housed in an accessible and easily deployed mobile system architecture. The framework provides a solution to current limitations that have been identified in developing existing food recognition programs within the context of available standards and proven design methods.

3.3 System Requirement

Requirements are one of the most critical aspects of intelligent software development, converting concepts into measurable technical and operational requirements. Requirement specifications, which outline the success of the many individual components that comprise the system as a whole, as defined at a high level by the System Framework, establish detailed capabilities and constraints performance standards that must be met for each component to meet its objectives.In software development, documented requirements are a basis for validating design, planning for implementation, testing, and scalability for the future. When creating a system to support visually impaired individuals in real time diet analysis, it is important that the requirements not only specify technical functionality but also consider practical usability, reliability, accessibility and deployability.

The proposed AI Augmented Food Analyzer system is intended to serve as an intelligent assistive system, which includes both functional and nonfunctional requirements. The functional requirement specifies what the system must accomplish and the nonfunctional requirement determines how well it must function at the practical level. Both the functional and nonfunctional requirements work together to ensure that the system will function technically as well as be usable, reliable and accessible to those in normal operational environments.

A major requirement of the system is to allow the user to take pictures of food via their mobile camera in a realtime manner directly inside the app, without needing external assistance tools preprocessing steps. The camera interface should be easy to use and accessible to all users, requiring minimal interaction complexity, while allowing users to quickly input visual data into the app. The aim of the app's camera interface is to accommodate the needs of visually impaired users and as such, it will include simple interaction mechanisms and guided workflows to provide less reliance on visual validation.

A second functional requirement of the app is to allow for accurate classification of food images captured by the mobile device, using deep learning models created using pre defined food categories classification engine. In other words, the food classification engine will take a food image as input, produce the most likely predicted food category corresponding to the image, and output the prediction in an organized manner. This is especially important because food classification is a fine grain classification problem where multiple food categories appear similar to one another, therefore the app's food classification subsystem is expected to demonstrate a strong discriminative ability for a wide variety of food categories and presentation methods. It is also very important that the classification system accurately classifies food inputs, as incorrect classification may result in misinformation regarding the food consumed and therefore inaccurate nutritional intake.

The third of the functional requirements is the ability to do food detection and food localization. Because most meals in the real world consist of many different types of food in a single picture or frame, the system needs to be able to detect one food object at a time, if it applies to what you are trying to accomplish. This functionality is achieved by integrating an

object detection framework that can provide multiple localized predictions of food in a given picture or video stream. If this wasn't available to the system, it would be limited to detecting single foods only. Therefore, the realistic composition or make up of a meal would be vastly misrepresented.

In addition to the first two functional requirements of detecting and localizing food, a major functional requirement for the application is the ability for the user to access retrieve nutritional information once the system has recognized a food item. When the system recognizes a food item, it must access what nutritional metadata is associated with that food item in a structured database, including but not limited to the approximate number of calories in that particular food, what the main ingredients are, and potentially any other dietary information such as nutritional categories i.e., grains, protein, etc. This final functional requirement allows the application to be more than just a simple recognition application, but will also create a meaningful dietary support application to assist individuals in making informed decisions regarding food.

Because the system has been designed for blind persons, it is important to have another basic functional requirement that has auditory feedback generated from recognition results and nutritional information converted automatically to structured speech output via text to speech synthesis. Users would receive information about the identity of the food and nutritional values audibly, without having to read the on screen content. This auditory interaction should automatically happen after the recognition process has been successful, and provide an identifiable and logical sequence of operations, enhancing accessibility to all users.

Along with functional requirements there are also numerous nonfunctional requirements, including performance efficiency, which need to be satisfied in order for the system to effectively meet its intended use. One

of these critical requirements is the performance efficiency of the application. The application must produce results in almost real time after an image has been captured so that the user's interaction appears natural and fluid. The presence of long delays in processing the images could potentially limit the usability of the application and create a disincentive to use the application again. Since this application is designed to be used in the real world as an assistive application, users should not have to wait for an unreasonable length of time to receive recognition results. Therefore, feedback generation and inference will need to occur within an acceptable amount of time, as determined by the requirements for interactive mobile applications.

Another significant nonfunctional requirement is reliable and consistent performance by the system. The system needs to produce stable and repeatable predictions given visually similar input data measured under similar conditions. Inconsistent predictions will lead to a loss of user trust, which can have a detriment to the user's ability to use the system. Therefore, the recognition pipeline must have consistent performance across moderate environmental variations, including lighting, viewing angle, image size, and background.

Another nonfunctional requirement closely related to reliability is robustness. The system must perform adequately under real world environments free of constraints rather than only providing acceptable results in idealized benchmark environments. Dining environments are often cluttered with visual artifacts including utensils, table surfaces, packaging, and decorative objects. Environmental lighting often changes significantly between indoor, outdoor, bright, and dim lighting. Therefore, the recognition system needs to generalize beyond training data provided from the idealized environment, and be able to perform at a high level in the practical

environment where it will be deployed.

Due to their target user group, accessibility and usability are still very important nonfunctional requirements. Because the interface will minimize the cognitive and interaction complexity for users by creating a simple workflow that allows them to conduct recognition with minimal navigation or manual configurations. To accomplish this, the interface will include large accessible controls, straightforward camera capture processes, and automatic feedback generation. In addition, following the accessibility design principles would require that the use of assistive technologies is to reduce not to add to the interaction burden.

Scalability is another critical requirement; therefore, the overall system architecture must be capable of supporting future growth without necessitating a complete redesign. Therefore, as food datasets increase, as increasing numbers of recognition classes are added, the system must enable the efficient retraining and extension of databases and the deployment of updated models. The modular system architecture and centralized backend will provide scalability by supporting independent upgrades of inference and database modules.

Maintaining the proposed system's server hosted deep learning models is also a major consideration. The backend system architecture must allow models to be replaced, optimized, and version managed without modifying the client side Android application. Separating these two concerns will reduce the long term maintenance and enable performance improvement on an iterative basis after the initial deployment.

The consideration of security and data privacy will be extremely important in establishing the security of any scope or systems developed with these projects. Over the course of the development of this application, data is sent to the inference server from the mobile application, meaning that

secure communication protocols must be established to provide security integrity and privacy to the users whose images you are capturing as they are transmitted to the inference server. While the application does not explicitly process protected personal information, security in the communication between the client and server is considered a best practice across modern applications. In addition, the overall architecture of the proposed AI Augmented Food Analyzer must also be feasible for implementation. The architecture must also be feasible for implementation with the available tools, computing resources, and deployment methodologies. This will include being compatible with the Android Studio as the integrated development environment for mobile development, the back end deployment framework for hosting the model and the manageable requirements for the server hardware for the inference.

Finally, the requirements for the proposed AI Augmented Food Analyzer will also outline the technical, functional, and operations specifications that are needed to successfully implement the system. These specifications will provide support for the performance of accurate food recognition as well as the efficiency, effectiveness, accessibility and sustainability of the system when deployed in a real world assistance context. Each specification is established to provide a definitive guideline against which the design, construction, and evaluation of the system can take place.

3.4 Feasibility Analysis

Feasibility Study As a critical phase of system planning, the purpose of a feasibility study is to determine if a proposed solution can be implemented within the technology, operational, financial and development constraints that exist. A requirement specification describes what a system is to ac-

complish; however, it is the feasibility study that determines if it is realistic or achievable with the technology, resources, and infrastructure which are available. In both the areas of software engineering and intelligent system development, a concept may sound like it is theoretically promising but it may prove to be impractical due to the computational demands, complexity of deploying the solution, cost to implement deploy the solution, or technical support required after the solution has been put into place exceeding the capacity that is available. Thus, a feasibility study is used to both verify that the proposed system activity is innovative and of sound design and theoretically feasible, as well as proposed to be practical and implementable as intended.

In order to analyze the AI Augmented Food Analyzer's feasibility consider it from numberous views as the system integrates deep learning with mobile app development, server deployment, database integration and user interaction focused on accessibility into one assistive framework. Since the project requires collaboration among mulitple disciplines the successful implementation of the AI Augmented Food Analyzer is based on both how feasible each sub module of the project is and how well they are integrated as a whole.

From a technical feasibility perspective the proposed system is very probable to implement due to the current availability and maturity of AI frameworks, pretrained models and development platforms. There are many examples of deep learning libraries such as TensorFlow and PyTorch that have integrated model training, fine tuning, evaluating and deployment capabilities in their own ecosystems. Examples of pretrained architectures include ResNet, EfficientNet, ConvNeXt, MobileNet and YOLOv8 exist in the public domain and are all well supported by the research community. That means that designing recognition architectures from scratch

will not be necessary and will reduce the level of implementation complexity. Instead, transfer learning and fine tuning methods of adapting existing architectures for the food recognition domain can be efficiently performed.

The benchmark datasets available e.g., Food101 and COCO offload a significant part of the technical feasibility by providing structured training data for food classification and background discrimination. Additionally, the existence of large scale annotated datasets reduces the burden of collecting and labeling massive proprietary training data from scratch, allowing the project to leverage existing datasets through data augmentation techniques, as well. Thus, methods for constructing the dataset can be used to increase the diversity of the dataset through data augmentation techniques, which diminishes the need to collect an overly large amount of data during the course of the project. Therefore, academic projects are not required to have extensive amounts of resources in order to conduct successful training using these existing datasets.

As is also the case with computational feasibility regarding model training, the availability of modern GPU based cloud and local training capabilities has resulted in a high level of practicality. Deep learning training that required enterprise scale infrastructure, such as GPU clusters, is now achievable using university provided GPU labs, dedicated workstations, or cloud based platforms, such as Google Colab, Kaggle or commercial GPU providers. Although deep architectures, such as ResNet101 and YOLOv8, are still computationally intensive to train, the current state of existing academic and development environments allow for successful training of these architectures as well.

The project architecture meets both commonly available software development tools and standard tools used within the software development industry. For example, Android Studio provides a robust and mature soft-

ware platform for mobile application development; it offers support for camera integration, user interface design, network communication, and text to speech capabilities.

The backend server can be deployed using commonly adopted frameworks such as Flask, FastAPI, or similar Python based deployment platforms that have existing connections to deep learning inference pipelines. As such, the ability to create deep learning models for use within this system will reduce the complexity of integrating these components together and provide a more efficient end to end development process based on existing technologies.

Operational feasibility refers to whether the proposed system has the ability to function and operate as intended in its intended real world operational environment while also meeting the needs of its intended target user base. In this case, the proposed system is operation feasible because it meets the overall practical needs of its intended target user group of visually impaired individuals that are seeking to maintain dietary independence. The use of a smartphone for deployment means that users will be able to access the platform through common and readily available consumer hardware, rather than having to acquire custom, specialized hardware. Since the majority of todays smartphones will have cameras, network connectivity, and sufficient processing network capability, users will be able to access this system without needing to purchase separate specialized hardware.

Additionally, the client server model increases operational practicality by distributing the computationally demanding inference from the device to a back end server, allowing the mobile app to be slim while still being able to utilize sophisticated deep learning models that are too resource hungry for use on the device itself. Consequently, operational performance

is sustainable even on mid range smartphones.

From an economic feasibility standpoint, the proposed system is extremely cost beneficial compared to traditional assistive hardware methods. Traditional assistive devices for blind individual's use require specialized hardware that often comes with high price tags. Conversely, the proposed system employs commodity smartphones and server infrastructure which significantly reduces costs associated with deploying the system. Since the classification models are software based, the majority of costs are tied to development effort, training resources, and hosting on a server, as opposed to manufacturing custom hardware. Thus, the solution is economically scalable, as well as having the potential for widely distributed implementation beyond the research prototype level.

The use of open source software frameworks, publicly accessible datasets, and pretrained architectures will reduce development costs by eliminating licensing fees and proprietary dependencies. In particular, this cost effectiveness is highly relevant in the context of assistive technology adoption, where the affordability factor greatly influences the adoption rate.

The combination of leveraging pretrained architectures, existing datasets, modular frameworks, and already established development tool sets provides evidence of the practicality of the proposed system from a scheduling feasibility perspective. An incremental approach to training, evaluating, applying, deploying, and integrating the system allows for the scheduling of manageable projects, as well as milestone based development.

Whilst the overall concept is feasible; however, there are still some practical limitations that need to be taken into account. First, if the user relies on a remote server for deep learning inferencing, they will need a reliable internet connection; otherwise, there may be times when they wait an extended time for a response due to poor connectivity. Secondly, if the

desired outcome is successfully accomplished using a server host, there will also be future maintenance requirements, including maintaining the server uptime, scalability considerations, and costs associated with the infrastructure beyond prototype level.

However, the constraints listed above do not diminish the overall feasibility of the system; rather, they represent manageable deployment limits that can be addressed by optimizing the system, scaling the infrastructure, and improving future architectural design, such as implementing a hybrid on device server inferencing system.

Overall, the feasibility evaluation shows that the AI Aided Food Analyzer is technically, operationalally, economically and developmentally viable given the project scope constraints and tool existing technologies available to develop the AI Aided Food Analyzer. The existing deep learning framework, pretrained architecture, benchmark datasets, mobile development environment and backend deployment technologies all combine to form a mature ecosystem which supports a successful deployment of the AI Aided Food Analyzer. Combining the use of commercial grade smartphone hardware and server based inference also adds to the practicality to deploy the AI Aided Food Analyzer while being economically accessible to users. This means that there is both conceptual value and practical value for the AI Aided Food Analyzer as an assistive application that will function correctly.

3.5 System Constraints

The proposed AI Augmented Food Analyzer is designed to provide functional and feasible solutions for visually impaired consumers. However, all intelligent systems have limitations to perform work because all engineered

systems are limited by the amount of data available to the system, the amount of computational resources available to the system, the environmental conditions in which the system operates, and the design assumptions made during the course of development. Therefore, identifying these limitations will result in creating realistic expectations about the system's performance, clarifying the operational boundaries of the application, and highlighting areas that may need to be changed or improved in the future. In the context of software and intelligent system design, recognizing these limitations, rather than being an indication of a weakness, represents responsible engineering practice since it signifies an understanding of both the realities of implementing these systems in practice, as well as the limitations imposed by the underlying technology.

As noted in prior research, these classifiers must rely on predefined classes of food that exist in the training dataset. This reliance on the training dataset's predefined classes leads to misidentifying when seeing a new food product; this misidentification sometimes takes the form of matching it with a class that is visually similar or not matching the image at all. Deep learning models create boundary definitions i.e., discriminative boundaries only for classes seen in the training data, so there is no conceptual data based understanding of categories other than those that were present in the training dataset. Therefore, the effectiveness of the system is limited by the collection of training data used to create it regarding the representativeness, coverage, and diversity of the data.

The aforementioned problem is compounded further by the potential for poor domain generalization of the model. When using food categories the model has been trained on, even with real world food products closely matching the model's training set, performance may suffer if the food and or presentation is dissimilar to the training data. These dissimilarities may

include variations in culinary style, geographical location of dish presentation, different ingredients, different plating styles or presentation methods, and different quality of imaging devices used to take the original photo. Therefore, although the system may perform very well on images of familiar food items, it may not perform as well on images of the same food item that are not an original representation.

Another limitation relates to environmental sensitivity. Even when using data augmentation or saliency preprocessing, recognition may be compromised under extreme environmental conditions. Lighting issues such as poor lighting, heavy shadowing, motion blur, over and under exposure, partial occlusion, severe camera misalignment, and cluttered backgrounds can negatively affect both classification and detection accuracy. Different from benchmark testing environments, real world deployment scenarios will have naturally occurring less controlled environments that may relate to reduced recognition accuracy.

Another limiting condition for the proposed system relates to the quality of user captured images. Since the mobile application requires real time camera inputs from the user, quality of the image captured by the user will be dependent on user position, steadiness of the camera, framing and focusing of the image, and environmental condition of the camera capture once taken by the user. For a user with visual impairments achieving ideal camera alignment will be difficult unless there are additional assistive mechanisms to assist the user. Therefore, user input variability will create an unavoidable operational constraint that may have a negative impact on system performance.

Another limitation of the nutritional estimation provided by the framework relates to the accuracy of the estimates. The framework is able to provide estimates of calories and ingredients of the recognized food category

being analyzed, but the estimates are derived from standardized nutrition database record entries that do not reflect true volumetric measurement of ingredients, nor do they accurately reflect a specific portion measurement of that ingredient. Because of this, the framework estimates average nutrition for the recognized food type, rather than the exact nutrition for a specific instance of the food being analyzed. In addition, since volume and weight of ingredients, ratios of those ingredients, cooking methods used and various ways of preparing each ingredient can have a significant effect on actual nutrition values, the estimated calories provided by the framework should be viewed as informative estimates, rather than exact quantities for diet purposes.

The other major limitation of the framework is that the framework relies on its backend server infrastructure to perform inference processing. Because the recognition models are hosted by a third party on a backend server and are not installed on the mobile device directly, in order to work the mobile app must have an active Internet connection between the app and the backend server where inference is performed. In environments where internet connectivity is poor, latency is significant or network outages occur, the response time will suffer or degrade to a point where usability is compromised. This inference model dependency provides a limit on how the mobile application can operate in offline, low connectivity environments.

Deployment based on servers creates problems of scalability and the limitations of infrastructure. The use of a backend host to improve mobile performance can only be enhanced by having all updates from a centrally managed location. As the number of users on the system goes up, it will likely need more server resources, more allocated bandwidth, and to be managed with more infrastructure than in the original prototype deploy-

ment. If the system was to continue to grow from prototype use and into larger deployments, it becomes critical to have backend capacity planning so that there will still be low latency for users who demonstrate a high demand on the system.

The system's limitations related to machine learning will include confidence uncertainty and ambiguity of prediction. Some food categories will exhibit more visual similarities than other food categories, but will likely be separate categories within the feature set. As an example, a dish prepared with similar ingredients; sauce preparation; methods of preparation and texture may have overlapping visual features making it impossible for a model to give an accurate prediction. Even when the model has been properly trained, you may have predictions made with lower confidence, or the classifier may have difficulty predicting between visually similar categories.

Another constraining factor for design comes from the way datasets are structured for training and recognition. Most of the publicly available food datasets use for food image identification have primarily single food items as their images e.g. for measuring performance against the common benchmarks. The type of food image that can be used in an object detection and augmentation strategy will result in some improvement in recognition, but because there are very few large scale benchmark datasets specifically created for recognizing different meal compositions as a context plate level contextual perturbations, the ability of models to accurately learn the complexities of plate level contextual associations will be limited. This lack of a large scale dataset for the recognition of meal compositions significantly reduces the ability of the model to recognize meal compositions across a wide variety of plate level meal compositions.

The other constraints are the more general ethical and dependency

issues related to the use of assistive AI systems. While the assistive AI application is meant to help users maintain their independence, the application is not intended to be used as a nutritional device that is medically certified or will be used in a situation with a medically critical dietary issue i.e. to replace professional advice related to diet for a person experiencing serious health challenges. The application is designed to be a source of information for users i.e. to assist users and will not replace clinical assessment of diet or diet allergy testing. Users must also remember that the recognition and nutritional outputs provided by the system are automated predictions and will have associated uncertainty.

The AI Enhanced Food Analysis System has a few limitations e.g. is based on trained food categories, does not contain an exhaustive database of foods to isolate poses research; limitations concerning light sunrise set i.e., very different intensity and angle, results can be unnatural due to own self respect respect -20%, irrespective of lighting; limitations regarding accuracy precision of nutritional output approximation; limitations regarding server connectivity; and issue of ambiguity with the food record on through camera that restrict the operation of the AI based system, therefore creating more realistic performance boundaries than available e.g., limited to no RI resource impact and clear methodology on how to adjust or retool the system in order to resolve these limitations; thus determining where future efforts need to be invested. As shown in Fig. 3.2, the proposed system follows a structured pipeline where the captured image is processed through detection and classification models, and results are generated based on a confidence threshold.

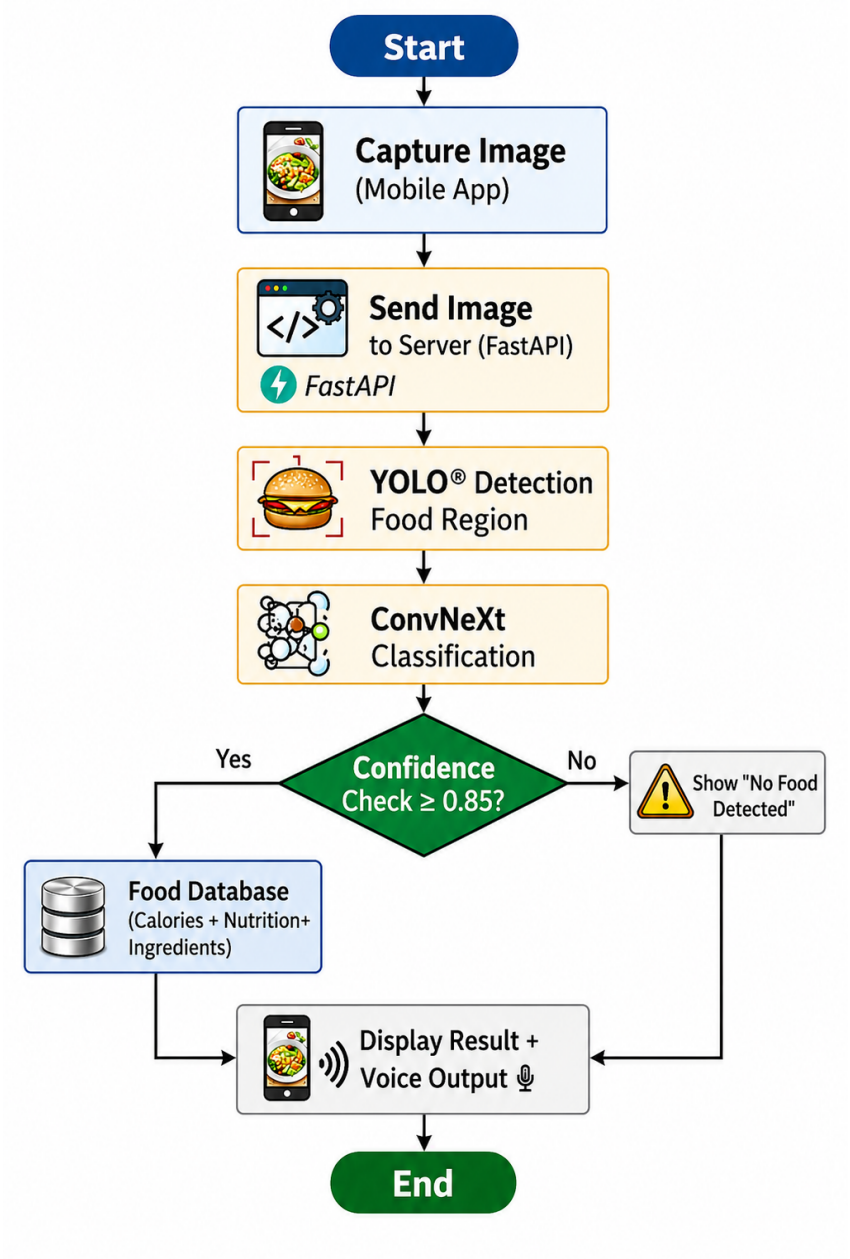


Figure 3.2: Proposed system workflow of the AI based food recognition system

Chapter 4

System Design

The system design is the last part of developing a Functional, Organized, and Structured System from the Conceptual Understanding gained from the previous chapters. In previous Chapters, the theoretical foundations, problem definition, and requirements were stated. This chapter now explains how those ideas can be organized in a manner that supports the development of a practical architecture capable of providing food recognition and analysis in real time. The AI Augmented Food Analyzer consists of multiple models that are organized into distinct, interconnecting components or "linked modules", which work together in a sequential manner. Therefore, the system design is focused on accuracy, but also on the efficiency, scalability, and ease of access of all components.

Incorporating modularity into the system development will allow for independent operation, as well as effective interaction with the other modules in the system. This design philosophy will enable changes to be made at one particular module e.g. improving a classification model or alternative saliency techniques without disrupting the functionality of the overall architecture. At the same time, creating a system that is lightweight, and quick responding is important in ensuring an effective mobile solution for blind users who require real time feedback from their mobile applications.

4.1 System Architecture

An intelligent food recognition system architecture requires an understanding that there is no single prediction of a food's identity or description that can create a successful perception of the food through visual information at the time of the prediction. Rather, the architecture provides a carefully planned progression of preprocessing and inference stages through which the raw visual information is then converted into a meaningful semantic

understanding of the food and it's characteristics.

To that end, an AI Aided Food Recognition and Analysis System, to be used by visually impaired individuals, will consist of a multi stage client server pipeline type architecture combining mobile image acquisition, back end preprocessing of the image, deep learning inference of the food from the image, nutritional information retrieval on the food, and generation of an auditory response message back to the user of the system. The architecture was intentionally designed for the purpose of supporting a balance between accuracy of the food recognition, the practicality of deploying the solution to the end user, responsiveness of the system in real time and the accessibility of the end user. While all of the processing is performed in the mobile application, the architecture of the solution takes advantage of distributing processing of computation between the mobile device Android app and dedicated back end inference server i.e., cloud in order to maximize system performance and maintain mobile usability.

The Android mobile app is the primary interface for user interaction with the system, as well as the primary method of capturing a food image using the phone's camera at the time of creation. The user will then take a picture using a simplified accessible interface designed for use by individuals who are visually impaired. After capturing the image, lightweight preprocessing tasks that include resizing the image, changing the format to one that is standardised, and compressing the image will be performed locally in order to prepare the image for transmission. This preprocessing step helps to ensure that all input dimensions are normalised, reduces the overall network bandwidth used, and eliminates the potential for latency from the time an image is captured until the image is transmitted. While limiting client side computations to lightweight preprocessing and the management of the user interface, the mobile application remains responsive

and efficient even on devices that have limited computing resources. Minimising client side processing is a common approach used in modern AI based mobile applications whereby all of the computationally intensive components of the inference process are executed on a centralised server to improve device scaling and maintainability.

The backend server of a system acts as a centralized intelligence engine and contains the academic Deep Learning Models that have been trained to recognize and analyze food, after the image has been preprocessed and transmitted securely via an API based communication layer on to the backend server. The Client Server architecture was selected due to the fact that most of the state only art Computer Vision Models that have been developed are very computationally expensive, especially when accurate classification performance at all times is required for mobile deployment in real time. By enabling inference to be offloaded to the backend server and by using more extensive and more accurate learned deep learning models on the backend server, mobile devices can perform as well or better than they did before without being limited in computational power from low battery life or memory space.

When an input image is fed to the backend, it will begin running through the visual perception pipeline. The initial phase of the visual perception pipeline is the optional saliency enhancement pre segmentation of the input image; this process will preprocess the input image to localise the food related area of the scene and exclude from analysis any surrounding or noisy areas of the input. In most cases, everyday food images will have other objects such as utensils, plates, napkins, packaging and table texture and or textures in the dining environment that will disrupt or degrade the ability to recognise food items or attributes. To reduce the clutter of objects and or textures typically referred to as background noise, preprocessing us-

ing segmentation inspired processes will highlight areas of the food related objects while diminishing the effect of nonfood related objects. The architecture rationale is based on semantic segmentation architecture principles established with such frameworks as DeepLab where a multi scale contextually aware feature space and dense feature extraction increases spatial resolution and region level awareness. By isolating the relevant food areas of the input image before classification, the system increases its reliability on noise from the environment and interference from the background.

Key features need to be separated from the image after going through preprocessing, and then the system can continue through the primary feature extraction classification pipeline.

Multiple pretrained large "deep learning" backbones ResNet101, EfficientNetB4, ConvNeXtSmall and MobileNetV3Large have been used in the construction of the system and have undergone an experimental evaluation to determine the best compromises between recognition accuracy and computational resource efficiency.

Each of these architectures offers a different set of benefits related to the overall recognition process. For example, the use of ResNet101 allows for precise and deep representation of residual features that will enable the capture of fine grain visual differences between very similar food categories by creating hierarchical residual representations from fine to coarse. The use of EfficientNetB4 provides a way to create CNN architectures that will allow for a balance of scale depth, width, and resolution and maximize classification accuracy and minimize use of parameters. Finally, ConvNeXtSmall offers a means of improving the feature extraction process through large kernel depthwise convolutional operations and architectural improvements based on the visual transformations of modern transformers, while at the same time minimizing the computational cost of a CNN.

MobileNetV3Large provides lightweight inference capability suitable for comparative evaluation against more deployment efficient architectures.As shown in Fig. **??**, the models achieve stable convergence during training.

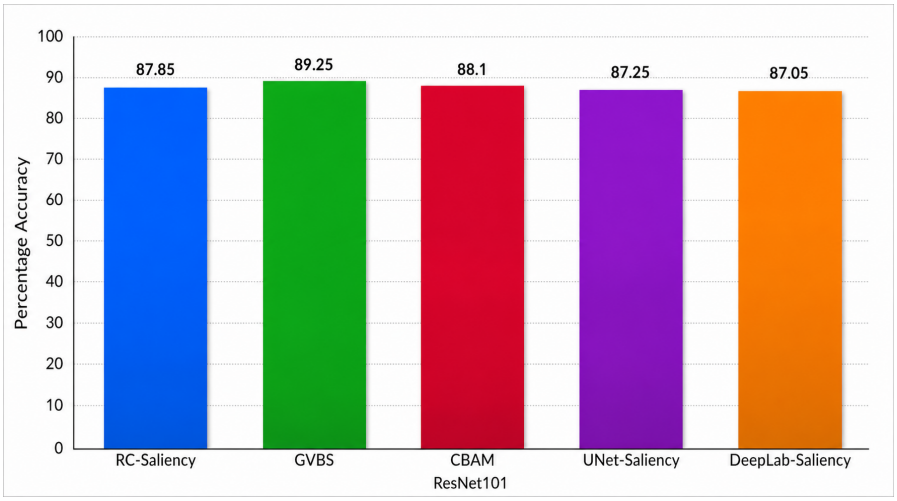


Figure 4.1: ResNet101

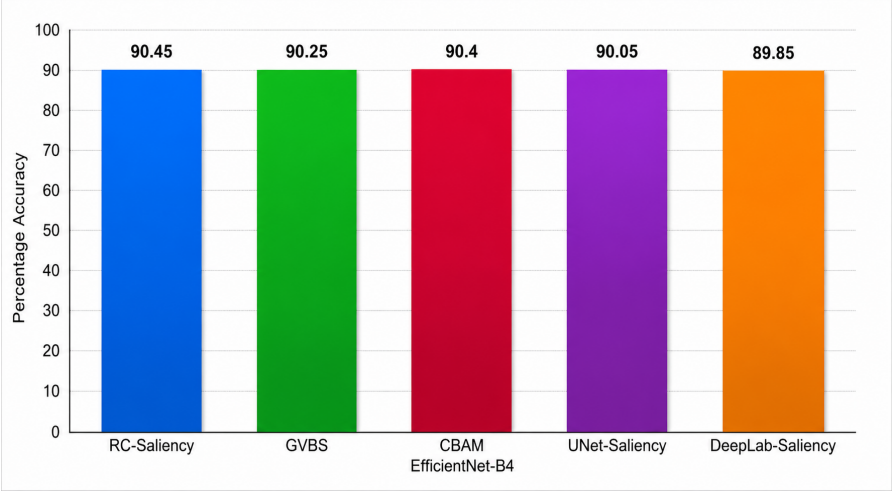


Figure 4.2: EfficientNetB4

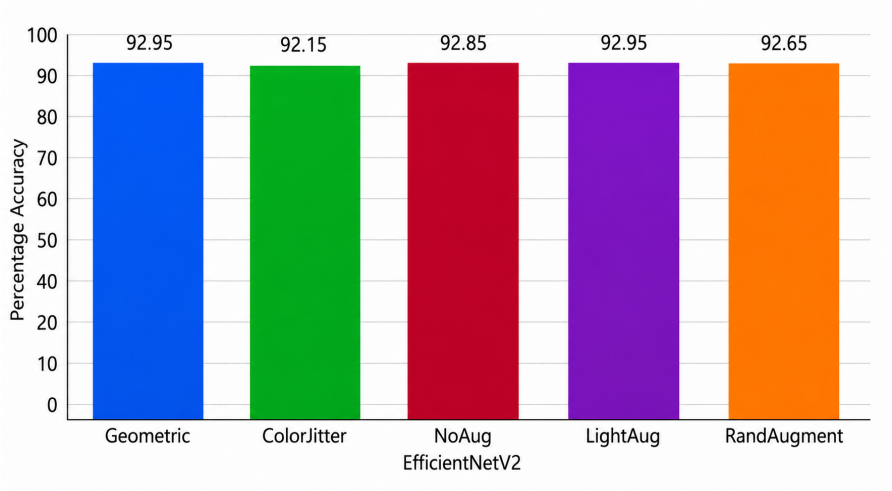


Figure 4.3: EfficientNetV2

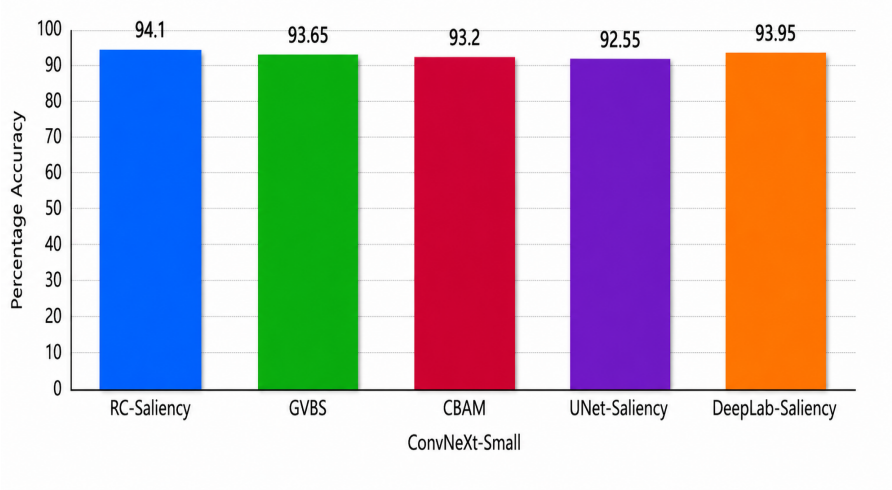


Figure 4.4: ConvNeXtSmall

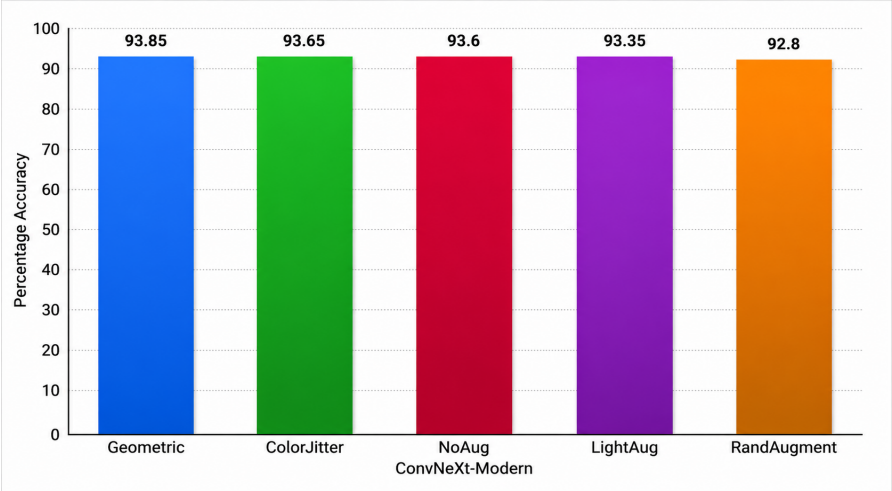


Figure 4.5: ConvNeXtModern

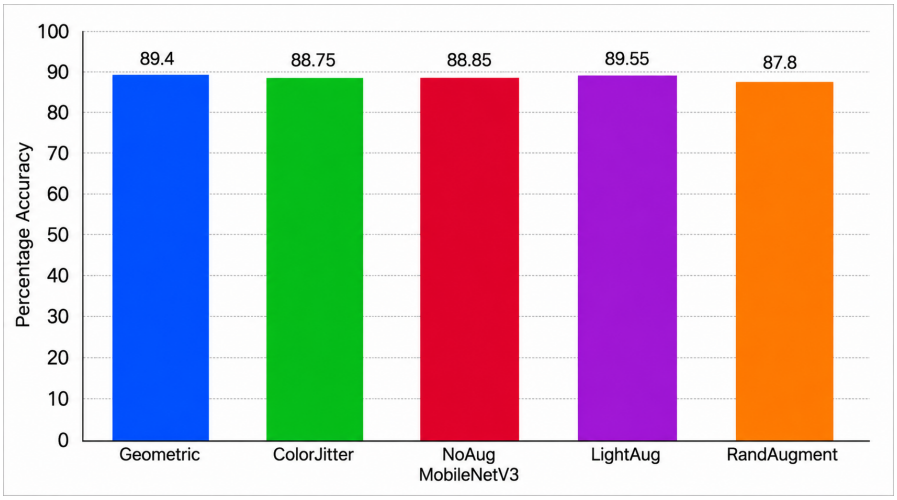


Figure 4.6: MobileNetV3

Table 4.1: Ranking of classification models based on accuracy

S.No	**Models**	**Accuracy (%)**
1	ConvNeXt-Small + RC-Saliency	94.1
2	ConvNeXt-Small + DeepLab-Saliency	94.0

Continued on next page

S.No	Models	Accuracy (%)
3	ConvNeXt-Modern + Geometric	93.9
4	ConvNeXt-Modern + ColorJitter	93.7
5	ConvNeXt-Small + GBVS	93.7
6	ConvNeXt-Modern + NoAug	93.6
7	EfficientNetV2 + RandAugment	93.5
8	ConvNeXt-Modern + LightAug	93.4
9	ConvNeXt-Small + CBAM	93.3
10	EfficientNetV2 + Geometric	93.1
11	ConvNeXt-Modern + RandAugment	93.0
12	EfficientNetV2 + LightAug	92.9
13	EfficientNetV2 + NoAug	92.9
14	ConvNeXt-Small + UNet-Saliency	92.8
15	EfficientNetV2 + ColorJitter	92.6
16	EfficientNet-B4 + RC-Saliency	91.9
17	EfficientNet-B4 + CBAM	90.5
18	EfficientNet-B4 + GBVS	90.5
19	EfficientNet-B4 + UNet-Saliency	90.4
20	EfficientNet-B4 + DeepLab-Saliency	90.2
21	MobileNetV3 + LightAug	90.0
22	MobileNetV3 + Geometric	89.7
23	MobileNetV3 + NoAug	89.5
24	ResNet101 + GBVS	88.9
25	MobileNetV3 + ColorJitter	88.9
26	ResNet101 + DeepLab-Saliency	88.8
27	MobileNetV3 + RandAugment	88.0
28	ResNet101 + CBAM	87.8

Continued on next page

S.No	Models	Accuracy (%)
29	ResNet101 + UNet-Saliency	87.5
30	ResNet101 + RC-Saliency	86.7

In the process of developing a reliable food classification model, one of the key challenges was not merely achieving high accuracy during training, but ensuring that the model could generalize effectively to real world scenarios. A model may demonstrate strong performance on a single train–test split, yet fail when exposed to new and unseen data. This limitation motivated the adoption of a more robust evaluation strategy K fold cross validation, a widely accepted technique for reliable model performance assessment in machine learning.

Unlike traditional evaluation methods that rely on a fixed division of training and testing data, K fold cross validation follows a more comprehensive and iterative approach. In this method, the dataset is partitioned into 5K equally sized subsets, known as folds. During each iteration, one fold is used as the validation set, while the remaining K1 folds are used for training. This process is repeated K times, ensuring that each subset is used exactly once for testing. The final performance of the model is obtained by averaging the results across all folds, providing a more stable and unbiased estimate of its predictive capability.we apply kfold cross validation test on highest achieving top 5 classification models. As shown in Fig. 4.7, the model maintains consistent performance across different folds. As well as shown in Fig. 4.8, the application of K fold cross validation improves the robustness and generalization of the classification model by reducing variance and providing more reliable performance evaluation.

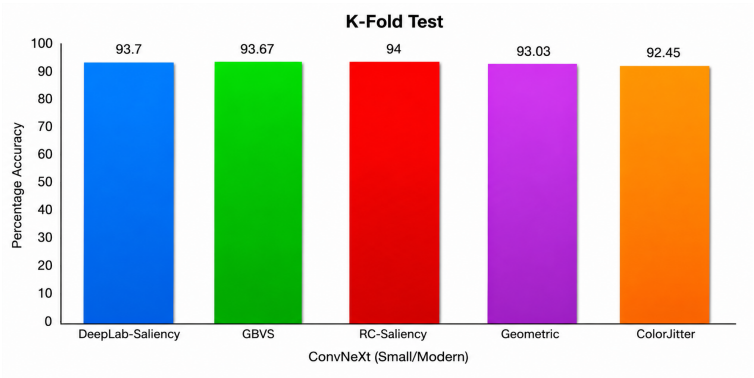


Figure 4.7: K fold cross validation performance of the classification model

Table 4.2: Accuracy After K Fold Cross Validation

S.No	Models	K Fold Accuracy (%)
1	ConvNeXt-Small + DeepLab-Saliency	93.70
2	ConvNeXt-Small + GBVS	93.67
3	ConvNeXt-Small + RC-Saliency	94.0
4	ConvNeXt-Modern + Geometric	93.03
5	ConvNeXt-Modern + ColorJitter	92.45

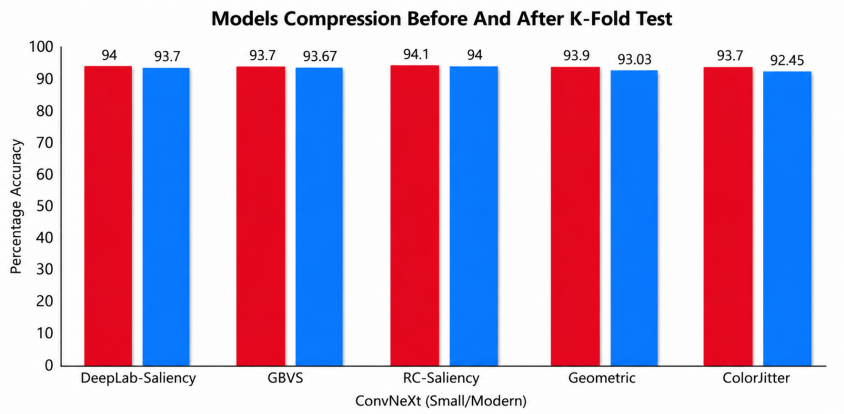


Figure 4.8: Comparison of classification models accuracy before and after applying K fold cross validation test

Table 4.3: Before vs After K Fold Accuracy Comparison

S.No	Models	Before K Fold (%)	After K Fold (%)
1	ConvNeXt-Small + RC-Saliency	94.1	94.0
2	ConvNeXt-Small + DeepLab-Saliency	94.0	93.70
3	ConvNeXt-Modern + Geometric	93.9	93.03
4	ConvNeXt-Modern + ColorJitter	93.7	92.45
5	ConvNeXt-Small + GBVS	93.7	93.67

The backbone classification model transforms the segmented food image into high level semantic feature representations through progressive hierarchical convolutional processing. Early layers detect fundamental visual structures such as edges, gradients, and textures; intermediate layers combine these into ingredient level patterns and compositional arrangements; and deeper layers learn highly abstract semantic concepts corresponding to complete food categories. This hierarchical feature extraction process mirrors the general CNN pipeline architecture commonly used in visual recognition systems and enables the model to distinguish between visually similar food items based on subtle structural and textural differences.

For scenarios in which multiple food items are present within a single captured image, the architecture incorporates a parallel object detection module based on YOLOv8. Traditional classification architectures assume the presence of one dominant object per image and are therefore insufficient when multiple dishes appear simultaneously. To overcome this limitation, YOLOv8 has been integrated into the overall architecture of the system as a real time object detection subsystem that can locate and classify several different types of food items within a single frame. YOLOv8's single stage anchor free detection pipeline allows it to perform bounding box regression localization and class prediction classification simultaneously with strong

real time performance. In addition, because YOLOv8 can detect multiple food items within one frame, it allows the system to perform beyond simple one label classifications and operate effectively in practical restaurant environments where several different types of food may be found together.

After successful completion of the recognition stage, the predicted food label is provided to the nutritional analysis portion of the system. The nutritional analysis module has an interface with a structured food database containing mappings from the recognized food categories to their associated nutritional meta data such as estimated calorie amounts, ingredient contents, and dietary information. The structure of the database layer provides rapid retrieval of nutritional information based upon the recognized food class as the primary key. The use of modularized perception and nutritional retrieval modules allows for improved modularity, maintainability, and extensibility of the entire architecture by allowing the nutritional information to be updated on an independent basis from the recognition models.

The final output, after nutritional retrieval, feeds into the response generation subsystem, which formats the recognised food name, ingredient list and estimated calories into a structured natural language response suitable for audio output. The client's Android device accesses the speech synthesis capability via its built in text to speech service, and therefore can provide immediate audio feedback for the user, without the user having to use their eyesight to see the output. The final stage of architecture converts raw AI predictions into an output that is useful to the user and represents the key means of providing accessibility for the entire system.

Another key architectural feature of the proposed system is modularity. The main system components consist of image acquisition, preprocessing, segmentation, classification, detection, nutritional mapping and

speech synthesis work as standalone logical modules within the greater pipeline. The modular architecture provides ease of maintenance and allows independent upgrades or replacements of individual component parts. For example, the recognition backbone could be upgraded to a more robust architecture in the future without redesigning the mobile interface or database layer. Likewise, the nutritional database can be developed independently of the perception pipeline. Such a modular foundation is consistent with the best principles of scalable intelligent system design and engineering.

Scalability is another major architectural consideration, as the inference will take place on a central, backend server, allowing the architecture to accommodate future increases in users by enabling horizontal scaling, load balancing, or multiple inference nodes to be deployed if there's an increase in the user base. Simultaneously, model updating will be simplified because retrained improved models can be deployed on the server side without requiring users to reinstall update their mobile application. Centralized model management is also a clear benefit of client server AI architectures in production deployment scenarios.

Overall, the proposed System Architecture makes the food recognition task part of an overall assistive perception pipeline instead of a singular classification task. By using a combination of accessible mobile interaction, server based deep learning inference, segmentation aware preprocessing, multiple model classification, object detection, nutrition retrieval, and speech based response generation, the Architecture creates a robust and scalable intelligent system designed specifically for visually impaired persons. The design is technical recognition performance focused, functional usability focused, modular maintainability focused, and realistic deployability focused. Thus, this integrated architecture serves to bridge the gap

between advanced computer vision research and practical assistive technology implementation.

4.2 Design Constraints

The design of a software system that is intelligent is determined not just by how it will work, but also by the limitations of how the system will work. Limitations that shape the boundaries of architectural decisions and how different components of the system are chosen, optimized and integrated into the system are referred to as design constraints. For the proposed AI Augmented Food Analyzer for Visually Impaired Persons, the architecture was influenced by many variables, including computational, operational, usability, architectural, and domain specific constraints. It is important to realize these constraints from the very beginning of the design process so that the resulting design can be realistic, deployable, and consistent with both technical feasibility and user centered accessibility requirements.

The complexity of deep learning model expansion was one of the major limiting factors in designing systems. Deep convolutional neural networks give significant visual recognition accuracy and have the capability of training millions of parameters, but the need for computational resources computing power to provide inference by the devices creates adding latencies storage and thermal constraints, and the result is that batteries will have their life shortened while being operated. Although smartphones have increased dramatically in terms of their capabilities, their performance is still constrained compared to their counterparts' GPUs and their respective data centers because of this constraint, it became necessary to establish a client server deployment model in which inference happens on a backend server for access and not at the 'edge' i.e. mobile device. The

common method of offloading implementations to backend servers is the norm for mobile AI artificial intelligence systems that cannot handle the direct complexity of deep learning models at the edge.

Server assisted inferring introduces an additional constraint based on compute limitations: the dependence upon the network for delivering inference results to users. While this improves the recognition accuracy of the visual representations by allowing for larger models to be available via the server's infrastructure, it also creates a dependency upon the availability of internet access to complete these operations. Accordingly, the overall design must take into account potential variations in network conditions and include designs that minimize communication overhead for best performance; thus, preprocessing of data was implemented with the focus of finding ways to create the smallest possible client side image payloads after resizing and compressing prior to sending them across network connections. Although these optimizations were implemented, they do not fully eliminate variation in latency due to the quality of the network, and so there will always be a tradeoff in the overall design between the complexity of the model being used for operational independence and the quality of service provided to users.

Food is not only visually complex in the world around us; it also presents a major challenge when it comes to recognizing objects food. While rigid object recognizing tasks have well defined geometrically shaped structures i.e. a box, food has many more constraints i.e. it is difficult to consistently recognize food as having a rigid structure. Food is complex in that it can have poor geometric consistency shape and also have very large variations due to the way that the food has been prepared, presented, and garnished within the same category of food and between different categories of food. The same food can look very dissimilar depending upon the way

that that food was prepared and can look very similar, even though it is from two or more distinct categories of foods. This ambiguity at food recognition level limits the maximum amount of accuracy that can be achieved by a food recognition system and also requires that the food recognition system implement architectures capable of discriminating between very fine grained features and that the use of saliency guided preprocessing as well as feature extraction techniques guided by attention within the architectural design would need to be included.

In addition, the variation in environment adds another significant design constraint; that is, the fact that dining locations in the real world are very unconstrained and can have different levels of illumination, cluttered backgrounds, shadows, utensils, dishes overlapping, reflective surfaces, and or parts of objects occluded hidden. In contrast to standardized datasets taken under relatively clean conditions, user generated images from real life generally feature many environmental artifacts that can vary significantly. Therefore, we cannot make any assumptions about an ideal input and instead need to design our system to be robust by using various mechanisms such as augmentation driven training, preprocessing using segmentation based techniques, and confidence thresholds. Although we have provided these approaches to mitigate some of the effects of the unpredictable environment, this variability is still an additional design constraint that will impose limitations on the degree to which we can consistently recognize objects within real world scenarios.

The needs of our user group impose further design constraints that are unique to this project. As the intended audience is visually impaired people, we cannot depend upon typical assumptions about designing visual interfaces to guide our design decisions. We will minimize interface complexity, dependence upon visual navigation, small size of touch targets, and

volume of text to the greatest extent possible. We will be providing simplified workflows for interaction; unambiguous audio output; and the least possible dependence on visual confirmation. The accessibility requirements placed upon our design may restrict our design flexibility, but will be critical to ensuring that our application is usable by our intended audience. In this context, accessibility is not a mere enhancement, but rather a primary design constraint that defines the structural makeup of the application..

The performance of supervised Deep Learning models is fundamentally limited by the amount of diversity, balance, and representation found in the dataset used for training. Although benchmark datasets such as Food101 have a lot of food images with labels, they do not represent all of the diversity that exists in food globally in terms of style, plating, or how it appears depending on the region. Furthermore, most food datasets consist of single dominant object images and not multi food meals. Because of this, model generalization was limited and as such, the original dataset was supplemented by COCO for COCO style background image richness and YOLO for practical robustness to improve the performance of food image classification systems. As a result, dataset limitations will still remain a design constraint with respect to food images that may limit the extent of recognition and generalization.

So, approximating nutrition involves a kind of architectural restriction. Nutrition may not be able to give exact weights, volumes or portion sizes because it's difficult to estimate these things from just a 2D image, and because of how the system has been designed to communicate when estimating nutrients after they have been recognized, which means that there is no way of accurately estimating how many calories were present in foods without the ability to perform volumetric analyses, portion measurements, and fine tuning accurate amounts of each ingredient. In order

for the system to operate with an acceptable level of accuracy, it has to use approximate nutritional mapping of food based on standardized databases of foods, which takes some limitation off the nutritional dimension of the application.

Another consideration is that maintainability required some design restrictions on the architecture. The intent of the project is to make improvements through new models, larger sets of data, and new categories of foods over time, so the architecture had to be modular and extensible. This design requirement required that design options be limited to loosely coupled components and API driven modular communication rather than tightly integrated, monolithic architecture. This design requirement added additional development complexity to the initial implementation; however, it resulted in increased maintainability over a long period of time.

Implementing security and privacy requirements added a third layer of constraints on our design. As users capture images of food and send them through the mobile app to be processed at the back end, we need to create secure communications between the mobile app and the back end that prevent intercepted communication or abuse of the images. While food images are typically considered to be less sensitive than other types e.g., people, implementing secure API communications proper handling of the back end for user image processing is still required for all modern connected application designs.

Lastly, the project resource constraints i.e., timeline and available hardware influenced the design scope. Since this is an academic capstone project, the design required balance between having lofty ideals while still being able to be completed in a defined time frame with available resources and technical complexity requirements. Based on balancing practicality with desire, many advanced features e.g., 3D portion estimation, multimodal

sensors, offline edge inference and adaptive contextual learning were not able to be achieved within the current scope of this project; therefore, a stable, complete and deployable core system product had to be developed using the restrictions that were provided by the capstone project.

To sum up, the proposed design for an AI Augmented Food Analyzer was shaped by multiple constraints that impacted it in multiple ways. Constraints include computational challenges of mobile devices, reliance on servers, ambiguous visual representations of foods in the food domain, variability in environment, limitations of available data set, accessibility, nutritional estimates that are accurate but approximate, considerations of how easy the solution is to maintain, security of any personal information collected and limitations of the resources available for the project. Although there were constraints that hindered development of the project, these same constraints provided the basis for architectural decisions critical to developing a realistic, balanced and deployment ready intelligent assistive solution. By recognizing and designing within these constraints, the project achieves an effective compromise between how technically sophisticated the solution can be, whether or not it is practicable to implement and whether or not it is user centred.

4.3 Design Methodology

The AI Augmented Food Analyzer was developed through a methodological framework of structured design. The end result was never simply a collection of modules but rather represents a complete and cohesive assistive platform that was engineered systematically. Design methodology is the name given to the structured process used to translate requirements into architecture in both software development and development for in-

telligent systems, continue to architecting into the build and then build to the final product resulting from validation of functionality as well. For such a complex multi disciplinary system as described above i.e., a food analyzer, where mobile development, backend deployment, deep learning technologies, and accessibility engineering are integrated with a database of items, a disciplined design methodology is critical to ensuring that development is conducted in such a manner that allows for rigorous effective data governance, modularity of the component systems, scalability of the entire system as well as for all functional modules of the entire project and to maintain traceability back to the original requirement throughout all phases of the development.

The design methodology selected for this project implements a hybrid iterative modular development methodology that is based upon both a combination of the software lifecycle development methodologies along with the development of experimental AI models. The reason for the adoption of this hybrid design methodology is that conventional linear methodologies are generally unable to appropriately accommodate for the continuous evaluation and refinement required for optimal model performance in AI based systems through experimental means. Intelligent systems require a series of iterative cycles of dataset preparation, model training, validation, tuning, and architectural adjustments prior to achieving acceptable performance while there is deterministic application software in which the behaviour of the application has been explicitly programmed. Therefore; for this project, a combination of the structured software engineering methodology along with the experimental machine learning workflow replaced the traditional software engineering development methodology.

The first step of the process was to analyze the limitations of current systems that help users identify food and provide nutritional information

to establish the overall scope and goal of the new method. In this step, the primary reason for the project was established, creating a method for identifying food and obtaining nutritional information specifically designed for blind users. Both functional and non functional requirements were established from this analysis, which were used as the basis for the architectural design of the system. A requirement driven design process is one of the core principles of structured system engineering since it ensures that the design decisions are consistent with the actual users' needs and not purely based on a technical exercise.

In the second stage of the software development process, known as the modular architectural planning phase, the whole system was divided into separate modules or units that do not depend on each other in order to improve maintainability of the system as a whole. The modules that were created to separate all different types of functions of the whole system included mobile client interface interface between mobile device of the consumer and the backend, image preprocessor module that prepares an input image for the backend to provide accurate predictions, inference engine the backend module that actually runs the classification of a food image, object detection pipeline the module that detects objects in an input image, nutritional database the module that provides information about the nutrition of all the foods classified, and auditory feedback subsystem the module that provides auditory feedback to the consumer.

During the third step of the methodology, the dataset preparation and preprocessing pipeline design occurred. The accuracy of any deep learning system is heavily dependent upon the quality of its training data; therefore, a significant amount of time and effort was spent preparing a solid dataset pipeline. After evaluating numerous datasets, the best option for classifying food categories was Food101. This dataset provides broad coverage

of food categories and contains a wide variety of food images. Negative samples from the COCO dataset were included as part of the training to improve background discrimination and therefore reduce false positives. The dataset preparation processes included normalizing, augmenting, balancing, and preprocessing data to establish a consistent format that can be used by the CDN for the model architectures selected to be validated during the methodology. Throughout this phase of the methodology, various preprocessing techniques were tested and implemented based on the experimental results of using them when building model performance.

The final phase of this project: Model Selection and Comparison. We knew that there couldn't be only one deep learning structure that would work best, so we tested all Convolutional Neural Networks (CNN) that are pretrained, using four different backbones—ResNet101, EfficientNetB4, ConvNeXtSmall, MobileNetV3Large. For each of these backbones, we transferred the learned weights using transfer learning techniques to make them suitable for food recognition. We then compared the models side by side to understand how well each one performed in terms of food recognition, processing, latency, and feasibility for use in real world deployments. The process of comparing models is a best practice when designing an AI system, as evidence should inform the choice of architecture instead of us simply making an assumption and using that to design the AI system.

Along with the classification portion of the project was the phase of creating a dedicated object detection and training process to recognize food. For our food object detection framework we chose the YOLOv8 object detection algorithm, which has shown to have great performance in terms of object detection in real time and has very accurate precise object location capabilities. The model used to train the detector consisted of datasets of annotated food object images, and the model was further trained us-

ing many data augmentation techniques, including mosaic augmentation, to enhance its generalizability to other food objects within different food scenes. The object recognition detector pipeline is built out independently but integrated into the overall structure of the food recognition framework in order to allow for flexibility operating under both single and multiple food recognitions at once.

Once the model was successful, the methodology transitioned from model development to designing the backend deployment architecture. A dedicated server side inference environment was created to host the trained models, expose prediction via API endpoints. The backend deployment layer was designed to receive images from the Android client, process those images through the recognition pipeline, pull nutritional info, and return structured predictions. API driven backend design was chosen, as it allows for modular communication, scalability, centralized model updates and creates a clean separation of frontend vs. inference logic.

The next step was to develop the mobile application and integrate the mobile application's interface using Android Studio. During this process, the Android client was developed to capture images, communicate with the server, provide auditory feedback to the user, and allow the user to interact with the application in an accessible manner. Specific attention was paid to simplifying access oriented workflows to reduce the number of visual dependencies of visually impaired individuals as they navigate the application. The mobile application was then integrated with the backend server to create a complete, end to end operational system.

The testing and refinement are the last step of the iterative methodology. The system was evaluated based on system performance i.e., classification accuracy, detection capability, inference delay, usability, accessibility, and end to end functional reliability. The weaknesses identified i.e.,

misclassification, latency bottlenecks, interface friction guided the systems iterative refinement for preprocessing strategies, model selection, threshold tuning, and workflow. The iterative evaluations provided feedback through the feedback improvement loop until the system reached acceptable and stable performance levels.

Another very important aspect of the methodology is that it enables continuous refinement versus a one time only implementation of a system. Due to the nature of intelligent systems, there is always a need for repeated experimentation, retraining, and optimization throughout the development life cycle. To meet these needs, the methodology was designed with multiple feedback cycles, as opposed to a strictly linear sequence of execution. Through the iterative adaptability of the methodology, balancing recognition performance, deployment constraints, and accessibility has been achieved.

The framework of this AI, AI Food Analyzer indicates a well structured, but flexible engineering process that is suitable for the heterogeneous requirements of modern intelligent assistive systems. This has been achieved through using a requirements driven software engineering process;decomposing the system architecture into modular architectures;selecting, developing and optimising an experimental AI model; developing and optimising an application that has accessibility as a core design principle; thus, the final product was developed to be technically functional but also systemically designed, able to be maintained and usable in a real world assistive environment. The disciplined nature of this methodology allowed for the creation of a functional intelligent application that was based upon articulate, conceptual project goals.

4.4 Database Design

The database layer of the proposed AI Augmented Food Analyzer is the structured knowledge repository that is responsible for converting the raw output of the deep learning model used for visual food identification into useful nutritional insights that can be accessed and used by the final user. The deep learning food identification system helps in visually identifying foods but simply identifying a food item visually using a computer program does not achieve the overall goal of assisting the user with their diet. In order to assist an individual who is blind in understanding the food they are eating, the identified food label must be linked to nutritional and descriptive information that includes calories, number of products, ingredients, and description of the food. The database design provides a connection between the information produced by computer vision i.e., identifying foods and nutritional insights about foods for the user, thus providing a semantic layer that connects the entire assistive technology system.

From a systems perspective, the database was created to enable fast retrieval, structured extensibility, and modular independence from the machine learning inference engine. The separation of data storage from perception was intentional to improve maintainability and flexibility. Storing nutritional data externally in a structured database allows the food metadata to be updated, expanded, or corrected independently of the recognition models. The modular separation of the data storage component conforms with software engineering principles governing the decoupling of the data layer from processing logic for improved maintainability, scalability, and long term system maintenance.

The database's core consists of a food entity mapping structure that associates each food category supported by the database with a unique

database record. This record, referred to as the canonical metadata entry for that food class, contains all relevant descriptive and nutritional attributes used by the application after recognition. The output label generated by a classification or detection model is used as the primary key for locating the appropriate record in the database. The mapping of label to record allows for efficient translation of structured dietary information from AI generated predictions.

To accomplish the supportive goals of this app, each food in the database has various characteristics attached. The food's name is stored as the first attribute; this name will be your main mode of identifying with the item, both while using the app on screen display and while generating sound feedback. Caloric information that is recorded in a food database is the second primary attribute of each food item, and provides an approximate caloric value based on nationally recognized definitions of nutrition; thus, an individual portion of a food and the way it was prepared can have differing molecular values calories. However, the approximate caloric content of most foods are listed in the app particularly per portion size standardized serving methods, enabling it to help with general dietary awareness.

The ingredient description is another essential field within the database and contains an open ended list of a food's common default ingredient classifications. This enables the app to provide users with more information beyond calorie counts regarding how a typical food is prepared i.e., by listing what is typically found in a common dish. For users who are unable to see, ingredient lists are also useful for identifying allergens other dietary restrictions, dietary choices preferences, and or what the recommended daily intake of a specific nutrient is. Other attributes fields that may be integrated into the database in the future include food category groupings, nutritional notes, dietary labels tags, common region definitions, etc.

Because food metadata relationships are relatively deterministic and structured, a relational schema model was used to create the database structure. Each food item exists as a record in a central food table, with the availability of supporting linked auxiliary tables if additional nutritional or category based metadata are required in future versions. A relational structure was chosen due to its high level of consistency, efficient indexing, and ease of integration with backend API logic. The scale and requirements of the proposed solution require relational storage as it provides a good balance of simplicity of implementation and management, as well as flexibility.

The database was optimized for runtime efficiency at a high level of read activity during operation. The food analyzer database primarily performs lookup activities post discovery; therefore, the database design prioritises the speed of response to queries and efficient indexing as opposed to providing complex transactional capabilities. Food labels have been indexed to allow for quick retrieval with a minimal latency of response time so that database access does not become a limitation of performance within the overall recognition response process.

The database can handle additional food types being added in the future, so no major changes will need to be done to the architecture of the system due to the database being capable of adding these new food types by simply putting records into the database and creating or updating recognition models accordingly. The overall goals of the design allow for the application's capacity to grow over time with the introduction of new cuisines, geographic regions, and other nutritional information. The separate architecture of the database and inference engine allows for the addition of food knowledge without needing to change the underlying backend logic, with the exception of updating existing recognition models.

Another consideration when developing the food recognition application was to maintain consistency between model labels and the database schema. Since the labels associated with each of the classes in the dataset created a numerical output the output of the neural network, you will need to create a matching entry in the database exactly equal to the label used in the model's output the model inference output in order to successfully lookup the food item as a recorded entry in the database. Therefore, we created a standard label normalisation scheme that ensured labels across each model, throughout the inference process, and from the database would match in order to prevent possible mismatches due to a difference in naming conventions, or a difference in how spaces or capitalisation were used to define a label across any of the various components of the system.

From a system operation perspective, the database resides in the backend server environment and will be programmatically accessed via API calls made by the application after the model inference process has been completed. When the pattern recognition engine finds a probable food type using its predictive model, it will instaneously execute a database search using the label returned from the recognition engine as a lookup key within the database. The nutritional and descriptive metadata returned by the database will then be combined with the returned recognition data, packaged, and returned via response to the Android client. The tightly integrated inference to database processing method allows for nearly real time ¡10ms transformation from visual recognition to the enhanced user experience via informative post recognition user feedback.

Integrity and maintainability were key factors in data design. All nutritional values, ingredient descriptions and, as much as possible, references to the various food groups were compiled prior to inserting them into the database in such a way as to ensure consistent representation between en-

try types i.e., one entry of chicken will have the same nutritional value description, regardless of the way in which it was prepared and consumed. Due to the potential for differing values between food reference databases, the values used in the current implementation were normalized and presented in such a way as to be suitable for "assisted estimation," but not to provide precise data for medical nutritional tracking purposes. This design provides an acknowledgment of the application's purpose as a source of assistance, rather than for direct clinical user nutritional measurements.

The current database schema will allow for future enhancements to the system's capabilities. Potential future enhancements include: calorie value adjusting based on the portion sizes of the food; tagging allergenic foods; allowing for more detailed breakdowns of macronutrients; supporting multiple languages for food names; supporting regional variants for specific cuisines; and providing personalized dietary recommendations to the user. The current database schema was designed modularly and to be scalable, thus allowing for these types of expansions to occur without the need for any foundational redesign of the underlying architecture.

Thus, the database schema as designed allows the proposed AI Augmented Food Analyzer to be the structured semantic backbone that enables the system to convert visual recognition data into meaningful assistive dietary information. By utilizing a modular, scalable, and efficiently retrievable relational database to organize the food metadata, the database is able to complement the artificial intelligence perception pipeline and allow the application to be transformed from a purely recognition based tool into a useful nutritional assistant. Furthermore, by integrating the intelligent perception of food via AI with a structured database containing precise and complete dietary knowledge, the overall system has significantly greater real world utility and assistive value for users.

4.5 GUI Design

The Graphical User Interface (GUI) for the proposed AI Augmented Food Analyzer has been designed taking into consideration the fact that it is primarily aimed at individuals who are blind; however visual interface design is also important when providing usability, accessibility support, maintainability, and user interaction flow to all users who may have some sort of difficulty interacting with a device or computer. When designing the GUI of assistive mobile applications, you should not just show information to the user; the GUI should also act at guiding users through their interaction with the application in such a way as to minimize confusion, decrease cognitive load and support integrating with nonvisual access methods such as with audio cues, and screen readers. Consequently, we approached our design of the GUI of our proposed AI Augmented Food Analyzer as more than just visual styling the GUI is part of the entire accessibility and usability design architecture.

The User Interface design philosophy is based on simplicity, clarity and efficient interaction. Unlike traditional consumer applications that tend to focus on creating highly visual layouts with many features, assistive applications need to restrict and carefully design their interfaces. Too many buttons, complicated menus with multiple layers and too much visual variety will create barriers to successful interaction will occur as many users will have either no vision or partially use non visual methods of navigation to perform their tasks. The User Interface was therefore designed to have minimal complexity while still providing complete functional accessibility through all the control elements on each screen and interface designed so only those screen and control elements that are necessary to accomplish the primary task of image capture, retrieving results and providing feedback

are included.

The features of the primary user interface screen were designed to serve as an interaction centre for performing the task of capturing food images for analysis. The layout of the primary interface has been designed to maximize the speed at which users will engage in food identification by placing the main camera image capture control in an identifiable location that allows immediate access. All other secondary interface elements including the navigation, setting controls and information are positioned to ensure that they will not detract from the ability to complete the main task of capturing food for analysis and will limit the number of clicks required to reach the point of completing the main user action of capturing an image of food for analysis.

The GUI was designed to facilitate an intuitive interaction sequence while providing a visually hierarchical arrangement of the GUI's components. Typically, larger controls, clear spacing, and consistent alignment of controls were utilized to direct users toward the primary actions of each control without risking any accidental selection of secondary controls. The hierarchical organization of controls also aids partially sighted users in maximizing usability, while at the same time increasing compatibility with the logic of screen reader navigations through maintaining a predictable interface structure. The research in human computer interaction (HCI) has consistently shown through multiple empirical studies that clear visual hierarchies and consistent organization within an interface have significant impacts on improving usability and decreasing the potential for user confusion relative to applications focused on providing accessibility solutions.

The typography, as it is presented via the GUI, was given careful consideration in the design of the GUI. To support partially sighted users, a large legible font was used in various types of controls, and strong contrast

ratios between the text and the background surfaces of the controls were used to provide greater readability under varying degrees of ambient light. To minimize ambiguity for all users, the labels and button text associated with the GUI were written using short clear phrases. While the primary emphasis of the system is to provide auditory output, it remains important to maintain visually readable interface text for the users who have partial vision, caregivers, and or supervisors and during demonstration situations.

Feedback visibility and transparency of the system state are also important components of a GUI. Users should be made aware of what the application is doing e.g., processing an image, waiting for a response from a server, experiencing an error by means of visual cues or messages such as loading animations, progress states, and confirmation messages that indicate to the user what is happening in the back end while their request is being processed. The presence of these feedback mechanisms gives users confidence that the system is active while they experience delays due to inference and makes them feel more at ease throughout this process. Transparency is especially important for interactive applications using AI, where processing will take place on the server side.

The result display screen is designed to allow the user to see the recognition outputs in an organized and easy to read manner prior to or concurrent with receiving an auditory indication of completion of the request via a vocalization of the result; therefore, all food names, calorie counts, and ingredient descriptions have been separated so they are displayed in distinct forms or areas of the display. This presentation results in visually reviewing results for partial sight users and allows for debugging, demonstration, and validation activities to take place while the system is being evaluated. The visual presentation of results will not replace auditory assistance.

The user interface design of the application had to take into consideration how people will interact with the app using different types of Android devices. Because there are so many variations in the size, shape, and resolution of Android devices, the layout was developed using responsive web based design principles. This helps provide a consistent look and feel of the user interface across many Android devices regardless of whether they are using a phone or a tablet. To achieve a proportional scale factor, the user interface employs flexible layout constraints and scalable components, which allow for proportional scaling of the user interface without breaking alignment or accessibility.

Another area of consideration for the application's user interface was to incorporate ways to handle errors and exceptions gracefully by implementing error handling and design for exceptional states in the user interface. For instance, if the application cannot connect to the backend server, if an image could not be uploaded successfully, or if the confidence level for recognizing an image falls below acceptable levels, the user interface gives the user visual and auditory indications of what has occurred. The application does not allow the user to experience a failed event or receive vague feedback; therefore, the user is clearly informed about why the event failed and what they can do to resolve the issue. Error handling and design for exceptions function as a critical characteristic of professional design on user interfaces, but they are also very important to assistive technologies users because they highly value clarity and reliability in their use of assistive technologies.

Standard spacing, standard button styling, standard icon style, standard font style, and standard method of interactivity create consistent interface screens throughout. A consistent design for the interface reduces the cognitive load on users the amount of effort it takes to think about

how to use something by allowing users to build on what they already know from one interface to the next, without needing to relearn how to use the interface. This is particularly important for accessibility focused applications, where predictability enhances users' confidence in how to use the product.

The graphical user interface (GUI) was created using the native Android user interface and layout framework via Android Studio, in order to ensure the product would be compatible, efficient and maintainable for the Android platform. Implementation through the native Android platform improves the application's compatibility with Android's accessibility services such as TalkBack, a text to speech screen reader, accessibility API's, etc., thus making the product more inclusive.

The GUI design of the AI Augmented Food Analyzer embodies the idea that usability should come before aesthetics in developing assistive interfaces. The interface is more than just a pretty view of how the backend operates; it provides an important channel for users to interact with the AI recognition engine. The GUI is designed for ease of use by adopting easy layouts, creating clear visual hierarchies, offering flexible designs that adjust to the user's device, providing simple but informative feedback, using legible typeface, and enabling users to identify when errors occur. By providing these features, the GUI helps users to easily utilize the AI recognition engine, as well as to have an easy time learning how to perform actions to enhance their ability to engage with the AI recognition engine.

4.6 Interface and Accessibility Design

An Assistive Intelligent System is determined by an individual's ability to utilize the algorithm completely independent of the system's ability to ar-

rive at the algorithm. Therefore, when designing an AI Augmented Food Analyzer for visually impaired people, the designers made sure that all aspects of the design were equally important. Accessibility was considered a part of the engineering process, rather than an enhancement to an existing system. Many standard methods for interacting with mobile devices rely on the use of a visual confirmation confirming a selection using a phone's display, menu navigation navigating through a list of options using a phone's display, and graphic display understanding a graphic image on a phone's display. Therefore, the interface for the Food Analyzer was designed around the principles of accessibility so that the system could perform as designed.

The foundation of the accessibility design philosophy is the principle of non visual operability. The design of the application was carried out in such a manner that the user could complete the complete recognition workflow with little or no visual feedback. The user is not expected to read any labels or look at the captured images, or navigate through complex menus; therefore, the application design incorporates guided interaction, simplified control flow, and automated auditory feedback. The overall design philosophy corresponds to existing accessibility engineering design principles which encourage a reduction of visual cognition and an increase of utilizing multi model feedback in the interaction of the visually impaired.

Text to speech output is one of the most important accessibility mechanisms incorporated into the user interface. This mechanism is the primary means of direct communication between the system and the user. After the recognition processing pipeline has been completed, the results will have been analyzed and converted automatically from the recognition result to a structured auditory output, which describes the identified food; an estimated total calorie value, and a list of the ingredients that are in the

food. Thus, by providing auditory feedback, the user can be provided with spoken information without the need to read from a visual display on the mobile device, effectively making the mobile device an interactive spoken assistant. The order in which auditory output is provided has been created such that the food will be identified first and then the dietary information related to that food; this structured delivery of auditory information aids in providing understanding of the information and also provides a reduction in cognitive overload that occurs when processing the information.

The design of the app has been intentionally created to create an app that has a simple interaction workflow. The number of user actions required to carry out the recognition process has also been minimized. In this way, there are only three user actions: (1) open the application; (2) take a picture; and (3) receive an on screen feedback message. Reducing the number of interaction steps minimizes cognitive load, reduces the learning curve, and improves overall accessibility. Complex navigation paths, multi screen workflows, and configuration heavy workflows create barriers for people with disabilities using the app and have been intentionally avoided in the app design because they create barriers. The app has been designed to ensure simple operation is a key element in the design of accessible systems.

The design of the app also made sure that any interactive element for touch interaction would be accessible for people with disabilities. For example, interactive controls e.g., capture buttons, navigation controls were designed with larger touch targets and larger gaps between them to ensure less accidental activation and improved tactile usability. Small dense controls are very hard to find using touch and or focus traversal with screen readers and as a result create frustration in using the app. By using larger touchable areas it allows more precise activation of controls resulting in

fewer accidents and lower levels of frustration when operating the app.

Designing with an emphasis on compatibility with all Android accessibility services was another major objective of the application's overall design. The interface was created to be integrated into native Android screen reader systems such as TalkBack. In order to ensure that corresponding accessibility APIs will correctly interpret and announce all of the application's interface elements, the button labels, content descriptions, focus order and interface interpretations, were all developed using appropriate standards for accessibility. This means that individuals who already have experience using Android smartphones and their associated accessibility tools can now have the same experience when interacting with the application as before, and they do not have to learn a new set of methods to interact with their smartphone.

The interface of the application also provides auditory feedback for operational state and confirmation cues so users are aware of the current state of the application as they use it. When users capture an image with the application, it will audibly or verbally notify users once the image is successfully captured, if recognition is in process, if recognition has been completed, or if recognition failed to occur. These feedback cues provide users with an understanding of what the current system state is without needing to rely on visual loading indicators or textual status messages. In applications that use AI assistance for back end inference where a processing delay is introduced, this feedback is necessary to hold confidence in the operation of the application and to reduce anxiety regarding whether the application is responding as expected.

The accessibility design included careful consideration of error handling in the event that the system does not recognize food with assurance or does not see food in the image or if there is a connection problem.

Users receiving clear and detailed spoken feedback about the error is much more helpful than the standard generic error messages that most systems provide. The transparent communication surrounding errors increases both user trust and usability since these users have knowledge of why the system failed and can take actions to correct it. In the case of assistive systems, ambiguous failure states can be confusing and erode a user's confidence in the technology when there is a specific way to communicate accessibility related errors.

A cognitive accessibility principle that has been strongly influenced by experience based design is the way design can create cognizant and predictable experiences. Designing systems that allow for cognitive accessibility must consider both sensory and cognitive disabilities and that interaction can occur in a mentally logical and predictable way. This is accomplished by consistent interaction patterns, consistent navigation logic, and consistent structure across screens. By using consistent structural patterns every user can learn and remember how the application works with a common layout that does not require users to constantly re evaluate the layout or learn different operating principles for different places in the application. Therefore cognitive efforts are lessened, and the ease of use over time is increased.

Future extensibility for accessibility improvements was taken into consideration when developing this interface in terms of the architecture of the existing platform. Therefore, other assistive mechanisms, such as voice command capture initiation, haptic feedback to assist in the positioning of the camera, multilingual auditory support, contextual dietary recommendations, and individual user preference settings could be added without needing to rebuild the core interaction flow.

In addition, the design of both the accessibility and interface were not

conducted separately from the recognition architecture but instead directly correlated with the backend system's functionality. Specifically, the timing of responses, the confidence and failure thresholds, and the output structure were developed to work in conjunction with the logic of the interface so users would experience all aspects of their interaction with the system from capture through to feedback as one coherent experience. The holistic integration of these elements into the system, rather than simply providing an additional layer on top of the recognition pipeline, means that accessibility is embedded into the system.

To summarize, the proposed AI Augmented Food Analyzer combines advanced computer vision functionalities with an interface and accessibility design to create a usable experience for people with visual impairments. Through non visual operational capabilities, automated text to speech feedback, simple file structures, large accessible controls, compatibility with Android's accessibility services, auditory state feedback, robust error reporting, and cognitive consistency when interacting, this system exists as a medium through which to provide a user friendly solution to those in need. By utilizing an accessibility focus as a primary design foundation rather than an afterthought, this project has successfully brought together both the worlds of research for artificial intelligence technologies and the practical application of assistive technology onto the market.

Chapter 5

System Implementation

The implementation of a system denotes the point in time at which the initial concept has been turned into an operational solution with all functionality included. The previous Chapters provided an outline of the conceptual framework and designed conceptual framework of a system. This Chapter outlines the implementation of the concepts defined in those Chapters and includes the methods used to implement models, integrate software and create a strategy for deploying the AI Augmented Food Analyzer as an AI based solution for the accommodation of visually impaired individuals. The AI Augmented Food Analyzer is an integrated solution which utilizes multiple AI components, including classifications networks, object detectors, saliency enhancements, and usability designs, to provide real time assistance to visually impaired users. The implementation of such a solution requires a method for coordinating how these multiple intelligent entities work together to facilitate efficient function and reliable performance in real world applications. In addition to achieving a high level of recognition accuracy, the AI Augmented Food Analyzer must also have minimal latency, be scalable, and remain easy for the end user.

5.1 System Architecture

In this chapter we describe the working architecture of the AI Augmented Food Analyzer System. An architectural design defines how a system will conceptually be organized into its components e.g., hardware and software while the implementation architecture defines how a conceptual design turns into an operational process where real world data is gathered input, processed in an intelligent way, and then results feedback are provided to a user. For an assistive technology application that leverages mobile computing, deep learning inference, nutritional data and an accessibility

driven user interface, the architecture of the system must provide more than just a connection between the software components. The multiple levels of computation must be coordinated in such a way as to provide an accurate, responsive, modular and scalable processing architecture. Therefore, a multi level processing pipeline design was created for the AI Augmented Food Analyzer solution, integrating image acquisition, preprocessing, image related processing, inference processing, nutritional mapping, and accessibility related feedback in one overall end to end workflow.

The architecture of the system is that of a Client Server Intelligent Inference Architecture. In this case, the Android Mobile App acts as a lightweight Client, which allows it to utilize the dedicated High performance computing (HPC) Backend Server for executing computationally demanding deep learning processes. By separating the burdensome processing associated with deep learning from the mobile device, we can use larger, more accurate models without compromising responsiveness or portability. Client Server Inference Architectures are often used in modern Artificial Intelligence (AI) applications as they help to: centralize large processing loads by separating them from the lightweight Client Mobile Device, and enable Clients Mobile Devices to access more sophisticated AI capabilities via Network Communication. The purpose of this architecture is to allow the Android App to remain responsive and consume a minimum of battery while taking advantage of server grade processing capacity (HPC) for deep learning inference.

The image capture for the operational pipeline starts at the image acquisition layer in which a mobile app takes a realtime picture of a food object using the device camera. The picture taken becomes the raw visual input to the recognition system. However, images captured in the real world do not typically display food in isolation or in ideal conditions.

Variability in lighting conditions, background clutter, table top texture, utensils, food packaging, and environmental noise can all negatively affect recognition quality if the images are processed directly. To solve this issue, the architecture contains both an image preprocessing step and a visual attention step specifically designed to enhance the input quality prior to performing deep inference..

One of the major parts of this preprocessing pipeline is using Graph Based Visual Saliency (GBVS) principles to help prioritize pictures by directing an individual's attention to them. The GBVS model described in Harel et al., 2006 predicts which areas of an image will carry perceptual weight in terms of being able to draw attention to other visual elements. The vision model developed is based upon creating preprocessing rules that give preference to visually salient food areas while reducing the impact of irrelevant background areas using GBVS methods. Preprocessing using attention based methods provides greater efficiency to the recognition pipeline because it captures attributes of images that can be used for meaningful recognition based on semantic criteria rather than features throughout the image. Since most food recognition problems are complicated because of the large amounts of clutter in food scene images e.g., restaurant dining rooms, using attention based preprocessing is particularly advantageous.

When the image has completed preprocessing, it will enter the deep learning inference layer which is the core of the system architecture the brain. The processed image will be routed through one of two separate inference routes based upon the recognition scenario: a classified based food recognition route for single food dominant images and an object detection based food recognition route for multi food dominant image scenes. This architecture was put in place to create a system that is flexible and also

strong across a variety of food presentation scenarios in the real world.

YOLOv8 based object detection serves as the primary object detection system in the proposed architecture for food analysis. It provides a unified real time object detection framework that integrates the localization and classification of objects in a single pass, providing a system with low inference latency and highly efficient object classification. The uniquely designed anchor free prediction method of YOLOv8, the decoupled detection head of YOLOv8, and the multiscale feature aggregation design of YOLOv8 all make it well positioned to detect multiple items of food in complex dining environments. In the proposed architecture, YOLOv8 enables the back end server to identify and localize multiple dishes or food parts in the same image, thereby providing additional functionality beyond very simple single label object recognition to assist in detecting multiple food types in an actual meal environment.

The inference architecture also has capabilities for hierarchical feature extraction and multi scale visual analysis using deep backbones and feature aggregation layers in the deployed models. These deep layers of the model convert raw image pixel data into progressively more abstract representations of semantic features, enabling learning of similar visual patterns associated with the characteristics of food including the texture, colour, shape, and composition. The significant advantage with this learning through hierarchical deep representations is that the characteristics of many food items can be similar and have subtle visual differences, thereby requiring sufficient data to provide an adequate amount of abstraction to allow discrimination among food items.

Following the completion of the inference process, predicted food labels are sent out to the nutritional mapping database query layer where recognized classes are compared against structured food metadata contained

within the backend database. This backend metadata relational database holds all nutritional descriptive information about every supported food category. For example, this would include types of calorie estimates, ingredient descriptions, etc. By implementing separate recognition logic from nutritional data storage, there is modularity and flexibility in the architecture so that food metadata can be modified or added to without affecting the recognition model itself. This decoupling from a modularity standpoint increases maintainability of the system architecture and future extensibility.

After the nutritional data has been retrieved and the output for the prediction has been processed, the final recognition result is going to be formatted by the response formatting layer into an appropriate format to be returned to the mobile application running on an Android device phone. The response returned back to the client is inclusive of the predicted food name, confidence score, estimated calorie count, ingredient list and any other descriptive metadata necessary for the client interface. The backend will create a structured API response and transmit this response through secure client server communication channels back to the Android application from which it originated.

Upon receiving the response from the server, the mobile application enters the layer of feedback and accessibility, where the output of the predictions will be displayed visually, and the output of the predictions will be converted into spoken audio feedback via text to speech synthesis. This last architectural layer guarantees that the end user receives the recognition intelligence in a manner that can be accessed, and it is in an assistive manner. Moreover, recognition results and nutrition information are automatically vocalized via text to speech synthesis, forming a bridge between the AI based backend processing and the real world accessibility of recog-

nition and nutrition information for the visually impaired.

Another significant architectural component of the implemented system is a backend architecture optimized for production deployment. Rather than treating the trained deep learning model as an isolated research project, the architecture adds the model into a production mode inference service pipeline that has API endpoints, preprocessing handlers, inference modules, database connectors, and response serialization logic. This service oriented deployment architecture follows the best practices for real world implementation of artificial intelligence systems, in which trained models are to operate as maintainable software services rather than as stand alone scripts. This type of architecture will provide a greater level of scalability, maintainability, and future deployment flexibility.

Future extensibility was incorporated into the architecture. Extensibility of the architecture allows for the addition of new models of recognition as well as multi lingual recognition, dietary recommendation engines, personalized nutrition analysis and portion size estimation, without complete redesign of the entire system simply by expansion of the current modular layers of the architecture. Thus, the extensibility of the overall architecture supports the broader goal of establishing a scalable solution for future enhancements of intelligent dietary assistance.

The system architecture for the proposed AI Augmented Food Analyzer was purposefully designed to provide a balance between computational efficiency, recognition performance, accessibility and practical deployment considerations theoretical versus real world applications. Through the integration of client server inference architectures, saliency driven preprocessing, YOLOv8 based object detection, modular backend deployment, structured retrieval from a nutrition database, and feedback mechanisms driven by accessibility, the system will support the development of an in-

tegrated, operationally viable, intelligent assistive platform. This overall architecture will ensure that sophisticated, visual recognition models will not be developed in isolation; rather, they will be integrated into a practical, end to end, real world assistance solution for visually impaired users..

5.2 Tools and Technology Used

Integrating many different software platforms such as deployment tools and various machine learning libraries, along with additional supporting technologies, were necessary in order to effectively implement the AI Augmented Food Analyzer. Since the overall system incorporates backend model inference development, database integration, accessibility related user interactions, and Android mobile application development, there was no one solution that could provide all necessary technology solutions; an ecosystem of specific tools was developed around several different technologies selected based on their compatibility and performance.

Adding to the complexity of this process is that by using only a limited number of pre existing technologies to represent the system, developing an infrastructure to facilitate the use of the newly developed techniques became an essential step of the overall process. The result was that the collective use of many different technologies allowed for the development of assistive applications that turned the conceptual device design into functional, marketable products.

The primary programming language used during the machine learning implementation pipeline of this project was Python. There were many advantages to using Python as the primary language, the most obvious of which is that it has become the leading language used in the development of AI and deep learning technologies today. The simplicity of its syntax

and the combination of Python libraries created a rich and available data source for developers when creating new models. The ease of readability of the Python syntax allowed for easier experimentation using preprocessors and better techniques for training models. The strength of the Python ecosystem made it the logical programming language choice for developers working on the back end and model related development tasks.

In order to train deep learning models and conduct experiments, PyTorch was used as the fundamental machine learning environment for the purpose of completing this project. The architecture of PyTorch is based on dynamic computational graphs and is highly versatile in terms of manipulating tensors. This makes it a perfect choice for performing experimental deep learning workflows and developing models that are primarily research driven. With its user friendly debugging, modular design, and wide range of pretrained models available, the implementation and tuning of convolutional neural network architectures (ResNet101, EfficientNetB4, ConvNeXtSmall, MobileNetV3Large) in PyTorch was done efficiently. Another benefit of PyTorch in the optimization phase was the ability to perform transfer learning, construct augmentation pipelines, and create custom training loops.

The Ultralytics YOLOv8 framework was used successfully to implement the functionality of object detection within this project. YOLOv8 is an all encompassing environment used to train, validate, and deploy object detection models based on YOLO. Annotation parsing, configuration for augmentation, model training, running inference, and exporting trained weights for deployment are all simplified by this framework. The modular nature of YOLOv8's API and training pipeline reduces the development overhead associated with integrating state of the art object detection functionality into your own application.

Several supporting Python libraries were used in the project including NumPy, OpenCV, and Pandas as part of its infrastructure to perform numerical processing, image manipulation, and scientific computation. NumPy provided fast multidimensional array manipulations along with tensor preprocessing support; OpenCV allowed resizing of images, normalising images, preprocessing, and performing camera related transformations; while Pandas allowed food metadata and nutritional datasets to be handled structurally in the preprocessing phase of preparing them for the database segment. As such, these libraries provide the lower level computational backbone and support for the broader AI and backend pipelines.

The Flask FastAPI architectural style of web framework was used to implement the backend inference and API service layer for exposing the trained model inference endpoints to the Android client. The backend framework provides lightweight HTTP API routing, request handling, image upload processing, and structured JSON response processing. The backend essentially wraps the trained deep learning models inside the API accessible inference endpoints to allow for the transformation of research grade AI models into deployable service components that can interact with external client applications in real time. Deploying AI using APIs is an accepted practise as one means of operationalising machine learning systems in production type environments.

The client application for mobile devices was created using Android Studio, which serves as an Integrated Development Environment (IDE) and is used to build the design of the Android app, develop its user interface, integrate APIs, access the camera, and implement text to speech capabilities. We chose to use Android Studio because it has the backing of Google, has support for the native Android SDK and provides extensive debugging capabilities. Additionally, it supports the different types of

APIs related to accessibility. Using Android Studio as the IDE allowed us to develop the mobile user interface effectively and provided us with the ability to integrate easily with hardware features of Android devices such as camera modules, audio output systems, and various types of accessibility services.

The mobile app was mainly built using Coco Objective based Apple Development Technologies which provide Native app performance and it was designed to work directly with Apple's accessibility frameworks. By developing a Native app, it ensures enhanced responsiveness, enhanced device interoperability and improved integration of platform level assistive technology i.e., TalkBack and text to speech services than many cross platform alternatives.

For database management, the project uses a structured backend database system to manage the food category metadata, calorie information and the ingredient data. The backend server provides rapid retrieval of nutritional information from the database layer after a successful recognition. We chose structured storage as opposed to hardcoded mappings in order to provide greater ease of maintainability, greater scalability, and greater flexibility for future expansion.

The software engineering workflow practices enabled both version control and iterative development management. The structured development of models is shown by the use of iterative development methodology; this allows you to track the revisions to a model or backend and the changes made to a model or application all throughout the software development life cycle. Since system complexity typically increases, providing significant value for long term project maintainability, having a structured model development approach has also helped reduce the risk of integration when creating the system

The project took advantage of GPU graphics processing unit accelerated training environments to allow the rapid experimentation and optimization of models developed using deep learning technology. Because of the large size of the proposed convolutional and detection architectures, the training process would have been extremely slow if CPU only based, especially during the iterative tuning process associated with hyper parameters as well as experimenting with augmentation of data. By using GPU acceleration in the training process, the training times were greatly reduced and a practical number of architectures and configurations could be experimented with in time for the project.

Another important technological component of the system is the integration of the use of Android text to speech services as a means of converting the result of recognition to a product that is presented as auditory voice output. The built in TTS service of Android will convert to an auditory voice output from back end structured text, thereby creating the primary assistive means of communication for blind users. The use of native TTS services provides for improved system speed and responsiveness since there is no reliance on external sources of text to speech as a means of conversion.

The project leverages many tools and technologies that are representative of a state of the art full stack AI application development pipeline throughout the entire life cycle of developing and deploying the model, and all parts of the system from model development through user interaction with the assistive food recognition platform follow modern best practices in AI engineering by maintaining a consistent tech stack and leveraging technologies commonly used in the AI industry. The project technology stack was designed to be scalable, maintainable, and able to be deployed in real world use to support users with disabilities to utilize advanced deep

learning research for accessibility benefits.

The proposed AI augmentation designed food analyzer system is fully integrated with an end to end AI engineering ecosystem that provides deep learning experimentation, deployment of backend components, creation of Android applications, and multiple tiers of database integration together with GPU accelerated infrastructure for training, as well as testing of the solutions on physical devices. Each subsystem can be developed, validated and integrated into the overall work product in a reliable engineering workflow due to its comprehensive production oriented nature. Creating reliability in the overall development of the application was critical in converting conceptual design of the system into the operational functional application assistance that received actual deployment in an operational customer environment.

5.3 Development Environment

The AI Augmented Food Analyzer required a development environment that was completely designed and constructed to be able to support multiple stages of intelligent system creation, including deep learning experimentation, back end deployment, Android application development, database integration, and complete system testing from beginning to end. Development environments in the contemporary interdisciplinary field of AI applications will generally cover several software and hardware platforms or tools rather than being limited by a single software tool or hardware platform. Instead, the entire ecosystem, including all computational resources, frameworks, libraries, hardware accelerators, testing devices, and deployment infrastructure that are used during an engineering cycle, comprises the development environment for the engineering project. Therefore, it is

crucial to establish an appropriate development environment in order to enable efficient experimentation, reliable implementation, and ultimately successful integration of the many different pieces of software and machine learning components associated with this project.

At the machine learning development level, the development of the project was done inside of a Python based deep learning experimentation environment. These environments are designed specifically to support model training, fine tuning, dataset preprocessing, augmentation, and evaluation work flows. The libraries and packages created in the Python programming language used the Python based virtual environment to maintain consistency with their libraries, while also isolating the experimental configuration from each other, thus allowing the Python based virtual environment to be used for two different projects. Isolating experimental configurations in separate Python based virtual environments is the best practice for machine learning systems. The reason for this is that it facilitates easy to reproduce all experiments and reduces the number of potential dependency conflicts, which will ultimately support a more maintainable experimentation pipeline.

The training and optimization of deep learning models were done on GPU accelerated computing systems so that it could be possible to practically experiment with very computation intensive architectures such as ResNet101, EfficientNetB4, ConvNeXtSmall, and YOLOv8. Training such large scale convolutional networks would have caused excessive experiment cycle times if only CPUs were used, thus limiting iterative tuning of models and allowing for limited ability to perform comparative model evaluation. GPU Acceleration has provided a means for rapid forward and backward computing during training, thereby vastly reducing the development time of models and allowing for feasible testing of multiple architectures and

augmentation strategies within the overall duration of the project.

The ML Development environment incorporated standard science and ML libraries such as PyTorch, TorchVision, OpenCV, NumPy, Pandas, and all other supporting utility packages necessary to perform preprocessing, augmentation, annotation handling, and evaluation metrics calculations. Collectively these data analysis and processing libraries create a complete AI experimentation environment that is capable of supporting the entire machine learning development lifecycle from preprocessing raw datasets through final exporting of trained models.

The backend of the project was implemented in a Python server side application environment; this included both the server and inference integration. The server was configured to host the models, and provide prediction functionality via the API. The server environment had all of the necessary components web framework dependencies, inference serving libraries, database connectors, request handling middleware, and serializer tools for real time communication between mobile device and inference engine. In addition, the server side environment was created to reflect the true production style deployment environment as closely as possible in an academic project.

For mobile application development, the project used Android studio as the primary Integrated Development Environment IDE platform for creating native Android applications; the Android Studio IDE was created with support for all of the features for mobile application development. Within Android Studio there were Android SDKs software development kits, emulators, device debugging tools, accessibility testing tools, Gradle dependency management, and other necessary development ecosystem features layout design, XML user interface design, real time device preview, performance profiling, and debugging. Because of the development ecosys-

tem provided by Android Studio, iterative development and testing of the mobile user interface was greatly improved.

To perform real device testing for Android smartphones, the Android development environment had also been adjusted to allow for emulator based testing as well. Although very useful during the early part of development, emulator testing lacks the benefit of being able to conduct real time interactions with devices, which were crucial in assessing how well the camera worked with the application, measuring network latency, testing text to speech output, assessing how well the user can operate the application using touch interfaces, and confirming that the application can be used in practical ways when operating under realistic environment conditions. The validation of responsiveness of the application and the quality of inferring workflows and quality of assistive interaction with the application were much more accurately validated with physical mobile devices than they were from using emulators alone.

A dedicated server back end to host trained inference models for deployment in a production API service level environment was created. This server back end housed all of the necessary components for the inference pipeline including the nutrition data, preprocessing modules and routing logic required for operational deployment of the system. This server back end was configured to receive and process images uploaded from the mobile device Android app perform inference on those images using trained deep learning models look up and retrieve nutrition data and return classification results in an organized manner. Deployment oriented testing had the benefit of being able to extend beyond the offline product working on the server only testing and measure complete end to end inference under true to life product deployment environments.

The database development portion of the project was integrated di-

rectly into the backend system and designed for structured storage and retrieval of food related metadata.

Database schema design, database population input, database validation, and database query integration occurred in parallel with backend API development so that the outputs generated from the recognition of food user defined data and the means in which to retrieve nutritional information from the database would seamlessly integrate interoperate.

Version control and iterative project management practices i.e., structures for project management throughout development were used throughout the development of the project to provide for structured refinement and rollback capabilities among all of the interacting subsystems associated integrated with this project.

Due to the highly complex nature of integrating the mobile application logic, the backend deployment code, the database schemas, and the machine learning pipelines, disciplined version management was necessary during iterative development and debugging to support continued stability throughout the development of the project.

One additional key component of the project structure was the inclusion of testing and validation frameworks. Specifically, dedicated evaluation scripts and test pipelines were developed to evaluate the performance of the models, the inference latency of the applications, the reliability of the API, the responsiveness of the application, and the end to end operation of the system as a whole. The implementation of this testing framework provided for the systematic validation of isolated components as well as the complete and fully integrated operation of the overall system during the entire development process.

The development environment intentionally enabled iterative experimentation as well as modular integration. Because this project involved

significant experimentation using different deep learning models; different preprocessing approaches; and different deployment configurations; the system had to allow for flexible swapping and retraining of models without disrupting the overall application framework. The capability to conduct modular experimentation was essential in comparing different architectures and optimizing the final recognition pipeline.

5.3.1 Application Architecture

The proposed mobile application has a distinct structure that serves as the physical basis for providing the user with AI Enhanced Food Analyzer's intelligent functions. While its models of recognition and back end services provide the system's mathematical computational capability, how these systems are implemented through architecture determines how well those capabilities work together, how well data travels through the system, and how well smoothly the entire recognition process is experienced by the user. In intelligent mobile applications, architecture is more than just a consideration of software engineering; it is a way to translate technical capabilities to create a functional and integrated product. While a poorly built architecture can include perfectly functioning AI models, it can ultimately fail due to poor integration, bad workflow implementation, and poor maintainability. As such, designing an application architecture was treated as a critical engineering phase in developing the overall proposed system.

This application has been structured in a modular and layered way so that it separates out the concerns of interface logic, application control, backend communication, and data processing operations. Separating these components improves the maintainability, readability, and long term extensibility of the system, as it allows for each component of the system

to evolve independently to avoid destabilizing the rest of the application. Rather than building the application as monolithic blocks of tightly coupled code, the application was broken into logical modules, with each module delivering a well defined subset of functionality. This modularization has been especially helpful because this project required a combination of several interacting technologies, including, but not limited to, Android interface management, REST API communication, AI inference requests, back end mapping of nutritional data, and generating speech based output.

The front end layer of this architecture contains the components for rendering the mobile interface, creating interaction with the user's screen navigation including managing the camera for input, and displaying user facing prediction results; i.e., the front end layer provides the user facing interface to capture photos of their food, request a recognition task, and receive visual and or auditory feedback in return from the application. As the primary consumer of this application will have visual impairment, consideration was made not only to functional correctness but also to accessibility, simplicity and minimal complexity throughout the front end architecture. The workflow of the application was developed with minimal steps while establishing consistent behaviors of navigation throughout the application.

The application logic and workflow coordination layer control all internal state transitions, validate user actions, advance the workflow through screens, and control the communication between the front end and back end subsystems beneath the presentation layer. The application logic and workflow coordination layer will enable user interactions to proceed in a structured manner i.e. An image must be captured prior to requesting inference, back end service must first validate whether a prediction has been returned before showing results to the user, and speech output will only be

initiated after valid prediction data has been returned maintaining the stability of the application and providing predictable behavior for real world applications.

An additional layer of architecture was implemented to manage all client to backend communication and was developed specifically for this purpose. This module is responsible for assembling captured images into the correct API request format, transmitting the request to the backend recognition server, receiving prediction results, parsing the returned JSON data, and forwarding the formatted result to the presentation layer. By creating a separate layer of communication between the clients and the back end, the overall system maintains a loosely coupled architecture and allows for greater ease of maintenance and ease of making changes to the back end in future.

The architecture of the back end is based on the client server model. The Android app is a lightweight client for user interactions while all of the computationally heavy AI inference i.e., using TensorFlow is processed on a remote server or cloud infrastructure. The decision to have the server perform AI inference stems from the fact that many state of the art deep learning models (e.g., ResNet101, EfficientNetB4, ConvNeXtSmall, and YOLOv8) require a large amount of computational resources and exceed many mobile devices' capabilities for real time processing. Therefore, by offloading the AI inference to a backend server, the app can remain highly responsive for mobile users due to their mobile hardware limitations, but still make use of high performance server class computing hardware to perform recognition tasks.

In addition to the fact that the back end uses client server architecture, there is another major architectural element that supports the back end functioning: a stateless request response model of interaction between the

client and the back end for handling recognition requests. Each request for recognition will be handled independently from any other requests; in other words, every uploaded image will be treated as its own independent piece of inference to be performed by the server without any persistent session context of previous requests. By using this stateless design, the back end will be able to scale up better to process the inferences for all users, provide fewer complexities for request management at the back end, and follows the current trend in the design of distributed API's.This will also allow the inference results produced by the server to be independent and reproducible without reference to any previous user behavior.

In addition to the above, the architecture implements structured feedback loops between processing and interface components for consistent and responsive user experiences. As images are uploaded and inferenced by the backend, users receive visual loading indicators and processing notifications from the application to indicate that the system is analyzing their submission. This type of feedback oriented architecture is critical in AI driven applications since machine learning inference introduces processing latencies that can lead to perceived application non responsiveness by the user. By providing explicit communication of system state during the entire processing cycle, the architecture increases user trust and clarity of interaction.

Error handling and fault tolerance were also part of the architectural design and are an integral design consideration. Validation mechanisms provide structured exception workflow approaches for handling unsupported or corrupted images, failed uploads, communication issues with servers, or unexpected backend responses to ensure proper user experiences and to avoid crashes within the application. User facing error messages were constructed to maintain a simple and clear approach to explaining

errors while avoiding unnecessary complexity of technical technicality and jargon in communicating user facing errors. Ultimately, proper fault handling architecture at the system level helps to create reliable products and significantly increases the user's perception of software quality.

When we were designing the architecture, one of our key priorities was scalability and future extensibility of the system. With a modular design of the architecture, all the various subsystems like backend endpoints, AI model selection, nutrition database integration, and accessibility modules can be updated or expanded as needed—without requiring a complete redesign of the entire application. This will allow us to incorporate future enhancements such as local caching of frequently used data, tracking individual diets, providing multilingual speech output, and expanding the number of foods in our database—without changing the overall architecture.

The architecture was also developed in such a way that multimodal interaction through the use of visual components in the UI, auditory output pathways and accessible interface logic is integrated into one operational workflow. By providing multimodal support to the architecture, it can be used by both blind and sighted users while maintaining accessibility first interaction design principles.

In general, the application architecture for the new AI Aided Food Analyser gives an excellent software framework for making intelligent food identification available on an actual useable assistive solution on a mobile device. The design of this architecture will help in achieving a modular, scalable, and robust software application. The architecture is created by implementing: a layered modular design model; division of clients and servers; separate communication modules; coordinating a structured workflow; making the processing fault tolerant; and integrating an accessibly

designed frontend. Therefore, the architecture of the new AI Aided Food Analyser will ensure that the application remains maintainable, responsive, extensible, and easy to use. By implementing this type of architecture, the AI Aided Food Analyzer project has successfully connected advanced AI inference technology at the server level to practical mobile usability by creating a continuous software solution using intelligent food identification throughout the recognition process.

5.3.2 Android Application Development

As a means of offering accessibility to those who are completely or partially visually impaired, the Android mobile app serves as the primary user interface for interacting with the AI Assisted Food Analyzer system and the intelligent recognition system powering food recognition and nutritional analysis of food products. In this case, the back end system performs all of the CPU intensive tasks related to food recognition and nutritional analysis, while the front end application provides visually impaired users with an intuitive way to translate the back end system's capabilities into an accessible, user friendly, responsive mobile application. This means that development of the application was not limited to simple design, but also involved the coordinated application of principles of mobile software development, user interface design that focuses on accessibility, communication with the back end system, and device hardware functionality into an integrated, assistive application.

The application was developed in Android Studio, the official integrated development environment for developing Android applications, because it offers broad usage of built in APIs, complex layout management capabilities, device emulation, performance profiling, debugging utilities, and many other features. Android Studio also offers the development environ-

ment needed to implement camera integration, networking communication, text to speech output, and accessibility functionalities while remaining consistent across the many different modern Android devices. Native Android development was selected over cross platform development due to improved performance, enhanced hardware integrations, and unbroken compatibility with Android Accessibility Services.

We started developing the application by designing its overall navigation and screen flow hierarchy. The user flow was deliberately designed to limit complexity and the number of steps to accurately identify food through our service. When users launch the application, they see a simplified main screen that allows them to take an image of their food right away. These design choices represent design principles that promote accessibility based on minimizing navigation depth and quickly accessing main features. The navigation flow is designed to provide predictable transitions between each stage i.e., camera capture, processing, and display results.Figure 5.1 shows the primary dashboard available in our application. This dashboard contains the most relevant functions by allowing users to scan food, track their history and find help. The user interface of the application is designed to allow users an intuitive way of navigating the application while also maintaining a simple user experience. The camera system of an Android application is very important for development. The camera is integrated into the application, allowing users to take pictures of their food as they are eating, so that they do not have to find images in their gallery or in a file on their devices. Capturing images through the application makes it easier to use the recognition feature of the application and is more practical, as users can analyze their food immediately after capturing it. In addition to integrating the camera system, we had to integrate the camera API as well, which included rendering a preview of the image, controlling the capture

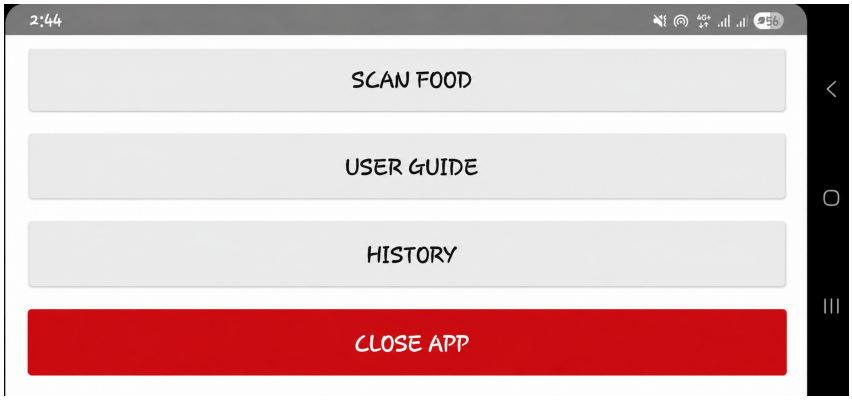


Figure 5.1: Application Dashboard Interface

process, managing the storage of the captured image, and handling runtime permissions that comply with Android's security requirements.Figure **??** shows the response from the proposed solution. The application was able to accurately identify food products and give appropriate details for each, including the ingredients in that product and an approximate total caloric count. The results of these outputs affirm that the AI based recognition system is a valid working model.

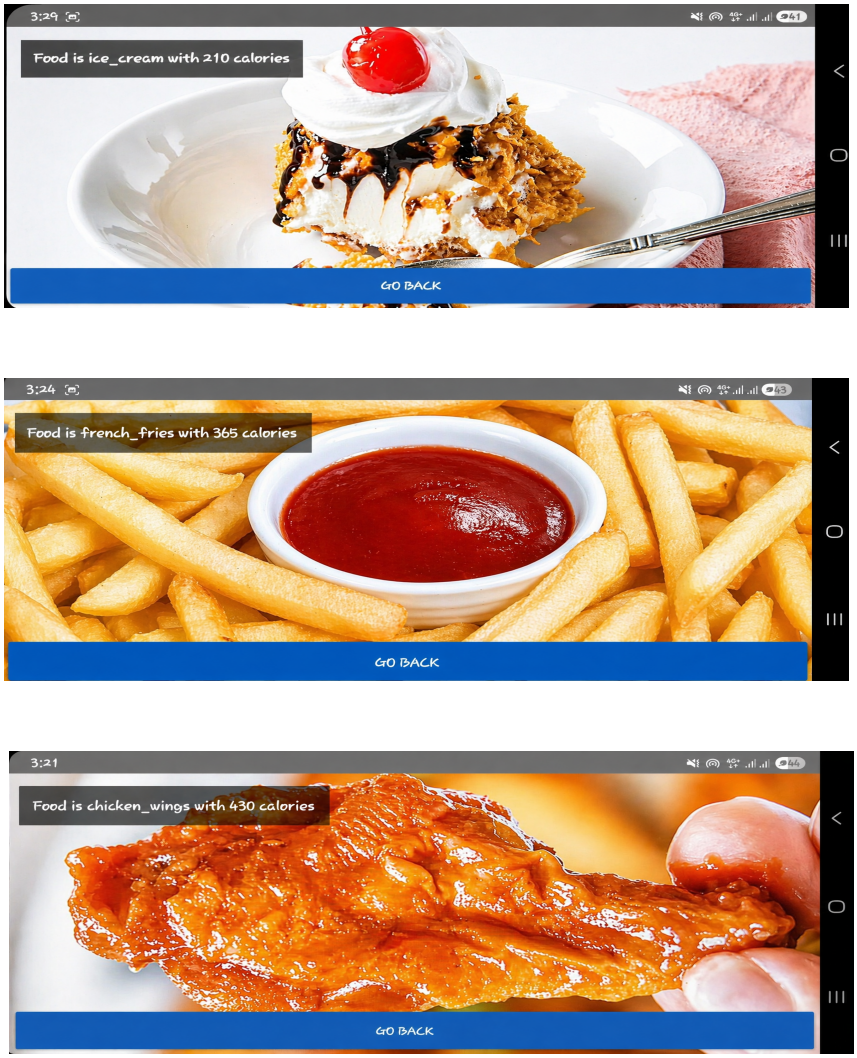


Figure 5.2: Food Recognition Results Generated by the System

The application performs local preprocessing steps on the captured image, including limited preprocessing of the captured image, before the image is sent to the backend for recognition at the server. This extensive server side recognition is processed on the server. Local preprocessing steps, specifically limited preprocessing of the image, help to increase the efficient transfer of the captured image by reducing both the size of the captured

image and standardizing the format of the captured image.

A dedicated networking component was developed to manage API based communication to the backend server. The networking component is responsible for converting and packaging captured images into multipart HTTP requests, transmitting the multipart HTTP requests securely to the inference API, processing asynchronous network responses from the inference API, and mapping the returned prediction payloads to application specific data structures. To ensure that non blocking backend communications occur while keeping the UI responsive during potentially latency sensitive inference operations, asynchronous networking logic is used. In Android development, maintaining UI responsiveness during network bound operations is considered best practice.

The Result Handling Subsystem of the Android app was created to provide the user with both visual and auditory representations of structured responses from the back end. Upon receiving forecasts from the server, the app will take information from the server response, such as recognized food labels, calorie estimates, ingredient information and confidence values, and display them within the Result Display Interface. In addition, the app will call on Android's built in Text To Speech Engine to sound out the Results for visually impaired Users. This combination of displaying results through visual and audio channels will be an example of a Multimodal Feedback Mechanism to enhance accessibility and to maintain functionality for partially sighted Users when in demonstration context.

Developing for Accessibility was a predominant consideration in implementing the interface elements of the app. The buttons were created so they have a large touch target area, adequate spacing, and simplified layout to enhance users' tactile use. All of the interface elements were given content descriptions and semantic labels that ensure compatibility with

Android screen readers e.g., TalkBack. The focus order and flow of navigation throughout the app were arranged to allow the nonvisually impaired to traverse the app in a predictable manner. Through these implementation considerations, the Android app will be usable for any visually impaired individual.

Structured loading and state feedback mechanisms were incorporated into the application in order to help users better understand the proceedings during backend inference operations. The fact that the recognition activity takes place on the server side requires an image to be uploaded for recognition and takes time for the inference to complete on the server side means that, while users are waiting for the backend response, the application displays loading messages and states for the duration of the inference process to indicate that it is processing an image. The application also provides optional auditory cues to indicate that the system is processing the image, which provides users with explicit processing feedback preventing them from assuming that the application is not functioning correctly due to an inference delay, so the overall perceived responsiveness is enhanced.

In the Android application, robust error handling has been developed to allow for seamless recovery from exceptional states resulting from all categories of errors including network failure, server down unavailable, invalid server response, and image not captured. Exception handling routines are used to provide understandable feedback to the user without impacting the applications ability to operate continuously due to the occurrence of any of these issues. Reliability oriented development has provided for higher levels of resilience and has ensured successful deployment preparedness.

Device performance and compatibility were also a major focus of the app's design and implementation. Android devices have a high degree of variability in terms of screen size, screen resolution, processing power, and

OS version number. Therefore, the application was developed using responsive layout constraints, and tested on a variety of devices with different sizes and configurations to ensure consistent presentation and performance in both hardware configurations. The use of responsive means for implementation has enabled the app to be more readily deployable and has provided the highest level of usability across the Android ecosystem.

Performance optimization during development also focused on minimizing unnecessary client side resource usage. Because backend inference already introduces network latency, the Android application was engineered to remain lightweight and efficient so that client side overhead does not contribute significantly to total recognition time. Memory management, image handling, and UI rendering were optimized to preserve smooth interaction even on mid range mobile devices.Figure 5.3 shows the history module, which stores previously scanned food items along with their identified ingredients and calorie information. This feature enhances usability by allowing users to review past results.Figure 5.3 illustrates how the history module works by storing foods that have already been scanned along with their ingredient and calorie information. Users will benefit from being able to go back and view what foods they have previously scanned, thus increasing user friendliness.

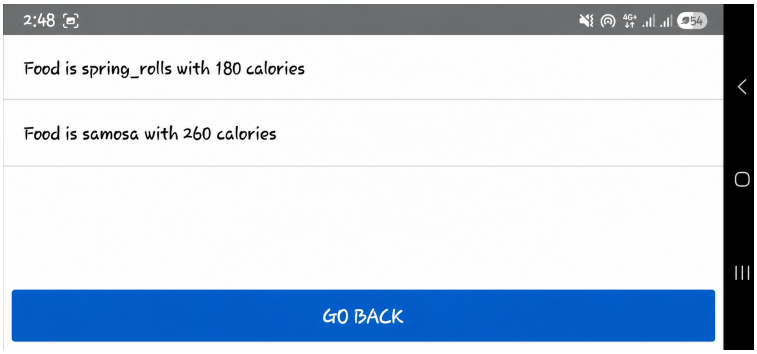


Figure 5.3: History Module Showing Previously Scanned Food Items

In summary, the transformation of the AI recognition backend into a fully functioning, interactive, mobile assistive system for real world deployment has resulted from the development of Android applications, which utilize native Android for the implementation of their respective apps. The app's front end functions include: Integration with built in camera capabilities; asynchrony in their communication with APIs; integration with multimodal approaches to presenting results; adherence to accessibility guidelines in user interfaces; structured handling of errors generated by users; and consideration given to performance. The front end of the proposed food recognition system provides a solid foundation upon which a food recognition system will be built. The app does not simply provide the user with an interface to view back end output but serves as the bridge between the user and the advanced artificial intelligence that will provide visually impaired individuals with access to the benefits of AI in their daily lives.

5.3.3 Server Side Implementation

The AI Augmented Food Analyzer's server side serves as the "brains" of the entire system, enabling the intelligent processing of raw food images into actual nutrition data. In addition to providing a user interface and image capture functionality via the Android based application, the server side AWS environment runs all components necessary to provide full functionality of the application. This includes running the "deep learning inference" pipeline, executing food metadata requests, performing preprocessing on food images, and hosting API endpoints for the application to consume and return to the user with nutrition data. Put more simply, the server side of the proposed AI Augmented Food Analyzer represents the connection between trained machine learning models produced by academic

researchers and real world applications for everyday consumers; thereby turning research projects into actual inference processing services that support everyday consumer usage.

Because of the high computation requirements posed by deep learning architectures regularly employed, recognition logic will be implemented on the server instead of the mobile device. Architectures such as ResNet101, EfficientNetB4, ConvNeXtSmall and YOLOv8 will require several million parameters to perform inference; therefore, execution of these models will require a large amount of processing power and memory. Execution of these model locally on a mobile device will introduce large amounts of additional latency, battery consumption, thermal load, and memory pressure compared to the mobile device's capabilities particularly, midrange Android devices. By deploying recognition logic on a server, recognition models can use a larger and more accurate model however operate in a responsive and efficient manner i.e., no perceived increase in latency on the mobile device. This is the deployment strategy universally employed in present day, AI powered mobile systems due to the fact that their computationally intensive inference workloads have typically been migrated to centralized computing capabilities.

The server side system was created using a Python based backend inference framework the PyTorch inference framework is included. This inference framework was selected because it supports model development via the PyTorch ecosystem. Because of this, the PyTorch trained weights, prepost processing pipeline development and inference code developed during the model development and training phase could be moved directly to the deployment environment with minimal conversion efforts. This reduces the complexity of the system engineering required for deploying a machine learning based system, while increasing traceability and making the system

easily maintainable.

The inference engine is responsible for executing the backend implementation, which loads trained deep learning models into server memory and makes them available for real time prediction. When a server is started, the classification and detection models that are chosen for inference will also be initialized and ready to use, allowing for the immediate processing of incoming requests by eliminating the need to reload a model before each inferences occurs. Preloading the models into server memory has a considerable impact on decreasing latency time per request and increasing the speed of response times for the deployed inference service.

Furthermore, an image processing workflow is included in the backend implementation that will standardize any incoming images prior to passing them into the inference engine. The Android client performs some light image preprocessing before uploading, but additional server side image preprocessing will be necessary to ensure that the images are in complete compliance with the input requirements of the model. This includes resizing the images to their expected input dimensions, normalizing pixel intensity values, formatting the image channels, converting the images into tensors and optional saliency educated enhancements or preprocessing transformations. To maintain the accuracy of the model during inference, it is essential that the image preprocessing done at training time and at deployment time are consistent.

When there are more than one food items being recognized, the backend processes the incoming images by sending them through the integrated YOLOv8 object detection pipeline for both localization and classification of various food objects in a single frame. Any detected bounding boxes along with the associated class predictions are subsequently post processed into structured response objects before nutritional mapping is performed.

For cases where there is a single food dominant image, the routing of circumstances allows the backend to alternatively route the input to the primary classification model, based on the operational configuration of the solution being deployed. This routing architecture provides the server with multiple modes of recognition while having a consistent API interface for the mobile client.

Once visual recognition has been completed, the backend calls on the nutritional mapping subsystem to query a structured food metadata database using recognized class labels that are returned by the inference engine. In addition to calorie estimates, ingredient summaries and descriptive food information about the recognized class will be retrieved as part of the metadata returned. The result of this stage, "Nutritional Enrichment," creates a useful dietary reference from the raw classification results.

All inference features provided by the server implementation are made available for use by way of a structured REST style application programming interface (API) enabling the Android application to access the backend via standardized HTTP requests. When the backend receives an uploaded food image via a specific API endpoint, it processes the image through the inference pipeline and returns formally structured JSON formatted responses that contain the inference result and the related information about the food item.The inference serving via an API method allows for a clear logical separation between the front end and back end, thus promoting the benefits of modularity, maintainability and scalability in relation to the design of deployment oriented systems.

In addition to the above, stable processing of inference request submissions and consistent formatting of backend response outputs were accomplished through the implementation of asynchronous request processing and structured response serialization. The inference prediction result data

is produced as machine readable JSON formatted response payloads that include fields for food label, confidence score, calorie count, ingredient list, and status indicator. Standardized response schemas improve the front end reliability of parsing the back end predictions, and will facilitate long term maintenance of client server integration.

Error management has also been integrated throughout the back end implementation to enhance the reliability and fault tolerance of each component of the overall system. Various conditions that could lead to failures, such as: invalid images uploaded, unsupported file formats, exceptions raised during model inference, database entries missing from the back end, and errors raised during the back end processing of an uploaded image can all be handled by the back end in an organized way, returning structured error response payloads, rather than causing uncontrolled crashes of the back end. The strong exception handling implementation will ensure that the server will maintain stable operation even when it encounters malformed or unexpected inputs.

In order to fully exploit the capabilities of the server side implementation for practical use, performance would also be a major consideration. As real time responsiveness is a must have to achieve practical usability, the backend was optimized to ensure inference latency was minimized through efficient model loading, execution in memory, preprocessing, and direct integration with the database for lookups. During development, inference timing and response serialization were both profiled in order to identify and reduce bottlenecks wherever possible.

The server side architecture has been designed to be modular in implementation so that each component e.g. preprocessing modules, inference models, nutritional mapping logic and API endpoints can be easily upgraded independently of each other in the future. Modular implementa-

tion allows the backend to be maintained and easily extended in the future with the addition of recognition model improvements and or expanded food databases.

In summary, by transforming trained deep learning models and the nutritional database into a deployable inference service with the support for real time assistive food recognition for mobile users, the server side implementation has enabled the central hosting of models, structured preprocessing, flexible inference routing, nutritional enrichment, API based service exposure, robust error handling and backend optimization focused on performance that will aid in the operationalization of the AI capabilities of the system. The server side layer will serve as the intellectual computation engine of the proposed platform and deliver a seamless connection from research grade CV models to real world assistive mobile deployments.

5.3.4 API Communication and Data Handling

Connecting the Android client application to the backend inference server in the proposed AI Augmented Food Analyzer is achieved through API (Application Programming Interface) communication. Although the mobile application and the server's intelligence are two separate computing systems in structure and function, users perceive them to be an integrated system when using the service as a whole. The communication mechanism used to send food images from the client application to the backend server and return predictions to the client application must be reliable, efficient, and well organized; only then will the client server architecture provide users with an integrated system. The API communication layer serves as the foundation for the functional architecture of the client server architecture, supporting distributed processing by allowing for interactions between the client mobile application and the backend server while main-

taining the responsiveness and usability that users expect from a mobile application.

The proposed system will implement a REST (Representational State Transfer) API communication architecture, whereby the Android mobile application communicates with the backend server through standard HTTP requests and receives structured responses formatted in the machine readable JSON (JavaScript Object Notation) format. REST was selected as the type of communication for the proposed system because of its ease of use, platform independent nature, widespread adoption, and compatibility with existing work flows from mobile to servers. A RESTful API is considered a lightweight and scalable method of providing communication between heterogeneous client and backend systems, which is important in designing a distributed application system.

To initiate communication, the user captures a food photo using the Android app and submits it for analysis. Before the request can be sent over the network, the app must perform local preprocessing and create an organized multipart HTTP request containing the captured image for transmitting via a network connection. After the image is packaged, the app sends the multipart HTTP request through the API communication layer to the predetermined backend image inference endpoint where the request will be processed. The multipart HTTP request format allows for the transfer of binary image data via a standard web service request format.

When the backend service receives the request, the backend API will parse the request payload, verify that the uploaded image is valid, and send the data into the image inference processing pipeline. After the inference and nutritional mapping processes are complete, the backend will create JSON formatted prediction output data for the incoming image. The prediction output data will contain food name, confidence level, calo-

ries, ingredients, processing status and error codes if applicable. Finally, the Android application will receive the JSON response, parse the response payload, and present the structured data in both visual and audio formats for the user.

The principal design goal of the API communication layer was to create a means for asynchronous and non blocking communications over the Internet. Since there is an inherent processing delay in performing artificial intelligence (AI) inference on the server side, network calls from clients cannot be made synchronously from the calling application on the main thread, or the user interface (UI) will become frozen and unusable. Therefore, to handle this, all API requests are asynchronously performed in the Android app so that network transmission and server processing will happen in the background, while the UI remains operational. Asynchronous communication methods are now mandatory for modern mobile applications that use remote services as part of their application workflows.

To guarantee that communications between the client and server are consistent, structured request and response schemas were developed. The request schema specifies how the uploaded images should be formatted and the types of content that will be accepted as well as any required parameters for the request. The response schema defines all of the data returned by the back end to the front end, and this is done in a consistent manner so that developers can readily parse the returned data from the back end. By maintaining clearly defined APIs, developers will be able to more easily maintain their applications, therefore, making it easier to debug and reducing integration errors during application lifecycle enhancements.

Handling errors and exceptions is an important aspect of the way communications occur through the API architecture. If not handled, network disruptions, incorrectly formatted requests, failing to process on the server

end, incorrect image formats, and exceptions on the backend will prohibit communication from occurring. Error detection and handling occur in two parts one at the client side and another at the server side using a structured approach. When errors do occur, the backend will provide an appropriate status code with structured error messages, which the Android application will interpret to provide user friendly feedback without exposing unrelated technical material. This also increases reliability and ensures that the system operates reliably, even in adverse operating conditions.

Another important aspect of API communication design is latency management. The user's perception of how responsive the system is, primarily depends upon how fast the backend can communicate. To minimize the amount of time spent in the request pipeline, all transmissions and serializations were optimized. Image prepprocessing on the client side shrinks down the size of images that are uploaded to the backend, while efficient JSON serialization minimizes the time involved in formatting the response prior to being sent to the client. These design factors help ensure that the interaction occurs in near real time despite the distributed nature of the architecture.

The design of communication through the API also considered security. The endpoints where the API's processing and interpretation occur were designed to provide only the least functional capability needed for reduced import into potentially threatening unsupported or malformed requests. The server instance that processes requests has validation and verification of the incoming payload prior to processing so that the server can continue to function as intended and in the same manner for its predetermined operational state. Finally, the architecture supports secure encryption of the channels used to transmit data from the mobile client to the server in the production environment.

There are other attributes that provide benefits for the API version of communication. The API defines the contracts that the two platforms the Android application communicating with the server using the API will honor. Because the API is the 'single point of truth' between both platforms, if either the client side platform or the server side platform is modified, it is not necessary to change or redesign either platform, assuming that the APIs remain the same. This design property will significantly increase the maintainability of all components the backend inference models, the server infrastructure, and the client interface design as each will be able to grow and change separately and do not require a complete redesign of the entire system.

The API layer also enables future extensibility for integrating with additional platforms in the future. Since all recognition related processing is exposed to the client application via standards based APIs, the recognition processing will be able to support other client platforms in the future such as Web applications, desktop applications, hand held data capture devices, and smart assistive technologies without having to modify the core inference processing model.

The API communication layer will allow the proposed AI Augmented Food Analyzer to operate as a complete and responsive intelligent system using its Distributed Client Server Architecture. The API Communication layer utilizes RESTful Communication, Asynchronous Request Processing, Structured Payload Schemas, Robust Exception Management, Latency Aware Optimisation, Secure End Point Design and Modular Frontend Backend Decoupling for reliable and efficient communication between the Mobile Client Application and Backend Inference Engine to enable Real Time AI Powered Food Analysis across multiple implementations of the same system. This Communication Framework turns all individual soft-

ware components into a unified Assistive Platform that can provide Real Time AI powered Food Analysis Implemented in real world deployment scenarios.

5.3.5 Integration of AI Models

Integrating AI models into an AI Powered Food Analyzer as part of the AI Augmented Food Analyzer project is the stage in which a trained machine learning architecture evolves from purely experimental components to functional parts of the assistive application that can be used in real world settings. While model training and evaluation provide the ability to recognize items through trained models in controlled environments, to enable a successful deployment of trained models in a real world setting requires the AI models to be implemented into a complete software solution that can receive live user input, perform inference consistently, and generate structured output data that can be ingested by the application. Thus, integrating AI models represented one of the most critical implementation steps since it moved the theoretical deep learning capabilities into the world of real world operational capabilities.

The integration of AI models into the AI Augmented Food Analyzer Project started by selecting the final set of trained models based on prior exploratory experimentation that had been completed in earlier phases of the project. This phase included comparative evaluations and experimentation across multiple architectures including ResNet101, EfficientNetB4, ConvNeXtSmall, as well as multiple lightweight mobile focused architectures to determine which architectures produced the best performing recognition results. In making final deployment decisions, having the ability to evaluate the performance of the AI architectures based upon four primary characteristics correct recognition of the item, speed of inference, foot-

print or size of the model, and deployment operational considerations was essential to make a final decision to assist in ensuring the deployed AI architectures provide not only accurate prediction performance but also are compatible with real time application performance requirements.

Once we selected our trained models we exported them from our experimental training environment into our deployable inference framework on our backend server. All of our model weights, configurations for preprocessing, mapping of labels and any inference logic was bundled into deployable inference modules that were compatible with our python based backend server. We were very concerned about maintaining consistency between our two environments training and deployment because even small discrepancies in preprocessing or interpreting labels would dramatically erode the functionality of our trained model when deployed.

The integrated AI pipeline can support both classification based and detection based inference modes therefore allowing diverse food recognition scenarios to be processed through the AI system. When only one dominant food is located within an image, the backend applies the deployed classification model to process through its predicted dominant class of food for that image as a whole. However, when there are several food types located within the same image; the system is able to localize and classify different objects independently using the integrated YOLOv8 object detection model. By implementing this dual mode of operation, the practical usability of the system can obtain additional capability to utilize the AI food recognition system, regardless of how the food item is presented during the meal.

One of the major difficulties that arise when working with AI models is that in order to deploy the model, there is a need to match up the model input prediction pipeline, with the model training configuration or

preprocessing requirements. The training of the model involves various steps such as resizing, normalizing, augmenting and transforming images based on the specific requirements of the architecture or algorithms used to train the model. To ensure accurate inference after the model has been deployed, the same logic that was applied during training was also applied in the inference pipeline in the backend or after the model has been deployed. The inference pipeline was able to at least resize the uploaded images to the model input dimensions, normalize the values of the images with the corresponding normalization constants, reconfigure the format of image channels as required by the model and convert the image data into the tensor format expected by the model when it was deployed. Ultimately for delivered models to perform reliably post deployment, consistency between both the training and inference preprocessing must be achieved.

After inference has occurred on the raw model output the raw model output is then post processed and interpreted prior to it being sent to downstream system components. For example, classification models use the output probability distributions generated by the model to determine the predicted class or type colour of object with the highest confidence score. On the other hand, YOLOv8 detection outputs will return information regarding the bounding box coordinates, class probability and confidence for each detection object predicted by the model. Hence, the model will apply its confidence thresholding logic to filter any potential false positive detections by lowering the likelihood that users will receive these results by suppressing those predictions that have a low confidence value.

The AI integration pipeline has a layer for label mapping. This is where the indexes or class IDs that the model outputs are transformed into human friendly food labels that correspond with the nutrition database and the application interface. Machine learning models output their predictions

numerically internally. Thus, this label mapping layer is how we take the predictions made by the model and convert them into meaningful semantic data that can be utilized by the rest of the application's pipeline.

An additional consideration in the AI integration work was optimizing the model performance; especially for deploying inference. While server side hosting provides more computational strength than running inference from a mobile device, it is still important to ensure the inference process is fast enough to maintain real time responsiveness. The following three optimizations were made to improve the deployed model performance: The model was loaded into memory all at once when the server was first started rather than reloading each time a request was made to perform inference, The processing pipeline was optimized for a minimal amount of overhead and The routing logic for handling inference was optimized to avoid going through unnecessary computation. These decisions helped reduce the total amount of end to end latency in making recognition.

The design of the integration architecture focuses on remaining model agnostic and modular, allowing new AI recognition models to be added and or replaced without the entire integration pipeline requiring a redesign. New models can be added by simply updating the inference module while still having the same API contracts and downstream data handling logic. The modularity of this architecture provides future proofing against changing integration requirements and encourages continued experimentation and improvement of models beyond the scope of the current project.

The inclusion AI of into the architecture of the service was also influenced by the operational considerations for the deployment of practical AI solutions. Since deep learning models are deployed from a central location on the backend of a web application, model updates can be made at the backend and deployed to the clients without requiring the client to

redeploy the Android application. As a result, newly improved recognition models can be made available immediately to all users via a model update at the backend, as opposed to being dependent upon users to manually update their client copy. The central deployment of models is a strong operational advantage of service architectures which are based upon the use of AI technologies.

From the perspective of assistive technology, the AI models which are integrated into the service represent the perceptual intelligence of the service; the computational component of the service allows the application to interpret food images, thereby providing independent dietary awareness for visually impaired users. If adequate AI integration into the overall service architecture were not provided, the overall architecture of the application would only function as an interface shell, without intelligent recognition capabilities.

The AI Augmented Food Analyzer project uses AI integrated into the food model machine learning network to convert trained AI algorithms into live operational inference services within the backend processing pipeline of an AI enhanced mobile food assistant. This involves using classification or detection models with structured implementation, replicating the preprocessing logic exactly, performing confidence based output filtering, mapping to the semantic labels, optimizing the inference, designing the modular architecture, and hosting the entire backend in a centralized manner. By bridging experimental AI research to deployable intelligent applications, the project successfully transforms advanced deep learning recognition models into practical mobile assistant functionality, thereby providing AI aided food recognition in the real world and within a mobile interface.

5.3.6 User Interaction and Accessibility Features

The effectiveness of an Assistive Mobile Application (AMA) depends on both the sophistication of its underlying AI technology and the ease in which users can interact and use the capabilities of the application in real world situations. As part of the overall implementation strategy for the proposed AI Augmented Food Analyzer (AAFA), user interaction and accessibility considerations were included during the application's design phase. Since the identified primary audience, i.e., visually impaired individuals, needed to be able to operate the system without having to make assumptions based on conventional visual interface design, user interaction workflows and accessibility mechanisms were required to ensure that users could access the technical intelligence of the AAFA back end system through a simple, intuitive and effective user interface. Therefore, a high degree of technical accuracy during the operation phase of the AMA was required in addition to an operational model i.e., how an individual with a disability interacts with the AMA that is accessible and usable for the end users of the AMA.

The application's interaction model has been purposely designed to minimize an operating sequence and maximize independence by way of a minimal number of steps within the operational workflow. Users will be prompted through a computerized process for image capturing, automated processing and providing feedback instead of having to navigate through each screen, menu or setting before starting to evaluate food products with food recognition: The application's workflow is simplified through this process, which reduces the cognitive load on users and minimizes the number of actions that memory needs to support for successful system performance. Thus, when developing mobile applications for accessibility, one of the de-

sign principles employed to generate a usable and learnable interface is to make interaction easier by simplifying the process of interacting with the application.

For users with visual impairments who interact with the application, Text to Speech (TTS) is integrated into the application as the primary output mechanism for receiving food recognition results. After the Back-end has analyzed the photo of a food product, confirming both the food's name and caloric content, and has determined whether or not any ingredients need to be added for a sufficient amount of nutrition; the food label, estimated calories and ingredient information will be articulated through the built in speakers of the Android device via the Android's native speech synthesis engine. As a result, users will receive complete feedback regarding the food that they have identified through food recognition via the device's display by hearing the result values verbally through auditory output from the speakers built into the device. The sequencing and phrasing of the verbal feedback associated with food recognition has been developed carefully to ensure that all information associated with a given food product is delivered in an orderly manner food name, calories and nutrients that can be understood by a recipient, and nothing is 'overloaded' with information when conveying the feedback as to the result of the food recognition process.

The tactile interface design of the app has taken into account accessibility. In designing the app, interactive elements and buttons were designed to have large touch areas, wide spacing between each of the buttons, and simple layouts to make it easier for the user to select the correct item by using touch and not look to select what they wanted. Very closely spaced or visually dependent controls can cause great difficulties for users with visual impairments trying to use the app; however, larger controls that can

be touched by the user will improve user accuracy and reduce accidental activations.

During the app's design, we included compatibility with the Android accessibility services. When creating the app's interface elements, all of the interface elements are tagged with content descriptions and have the appropriate sematic labels to allow Android screen readers e.g., TalkBack to interpret them correctly. We also designed the app to allow users using screen readers to see the layout of the app in a logical, predictable way when navigating through the app. By maintaining compatibility with established platform accessibility tools, visually impaired users can use the same assistive workflows as they would with a regular smartphone when opening the application.

Throughout the recognition process, the application gives users general interaction state feedback so they stay informed of the system's status. For example, after the image has been captured with a camera, the application may issue either auditory or visual confirmation that the image was successfully captured. While waiting for the backend to process the image after it has been captured, the application will provide either a loading indicator or an optional spoken cue so that the user knows the system is processing the image. When the results of backend processing are available, the application provides an explicit on screen message to indicate that it is transitioning from the image recognition part of the process to the feedback output part of the process. These mechanisms for providing state information during operation will help to alleviate users' uncertainty when using the system and assure them that the system is working correctly when there are delays in processing.

Error reporting has also been designed with an eye toward accessibility. If no food item is detected, the achieved confidence level drops below the

minimum acceptable threshold, an image is unable to be processed due to backend communication failure, or an image is unable to be processed by other means, the application provides the user with clear and descriptive error messages explaining the problem. Instead of providing users with vague or technical error messages, the application uses language easily understood by users to describe the problems to users in a way that leads users to the appropriate actions to correct the problems. The use of clear and accessible error reporting is particularly critical in assistive technologies. If users do not trust that a system will work correctly after they experience an ambiguous error, they are unlikely to continue to use the system.

Consistency in interaction patterns throughout the application was another key design principle. Buttons, screen layouts, navigation flows, and feedback mechanisms have all been designed with a focus on predictability and consistency — in order to reduce cognitive load to users i.e., the effort required to learn how to operate the system and creating a more seamless transition of operational familiarity from one part of the application to another. The relationship between consistency and improved usability and accessibility in human computer interaction research is well established; therefore, creating a predictable interaction model was critically important to building an application that delivers a high quality user experience.

The application was also designed to provide multimodal access to the interface. To accomplish this, auditory feedback, tactile interaction optimization, visually accessible layouts, and screen reader compatibility have all been integrated into one cohesive interface solution. As a result of this multimodal approach to accessing the interface, the application can be used by users with varying degrees of visual impairment, including those who are totally blind, partially blind, and those who demonstrate mixed

abilities.Figure 5.4 illustrates the user guide interface of the application. This portion of the application helps users to understand how to effectively perform operations within the application, including scanning food items and analyzing results.

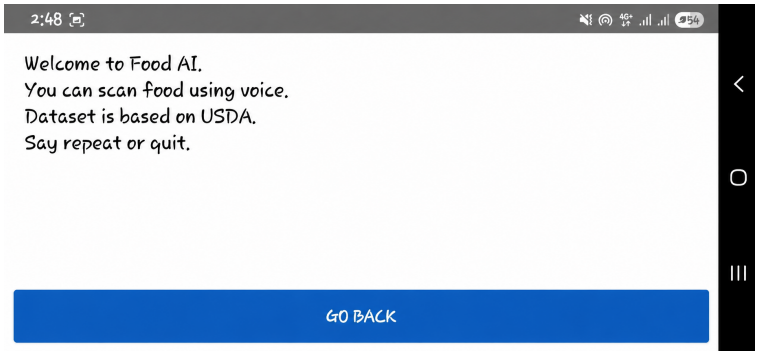


Figure 5.4: User Guide Screen of the Application

The key point here is that accessibility implementation was part of the design process from the very beginning rather than being added on after the fact when the functional part was done; designing with accessibility as one of the building blocks rather than an afterthought created synergy between the Interaction Logic, Interface Design and Feedback Mechanisms with the back end Food Recognition Functionality. Using this holistic design approach provides higher levels of coherence and quality in the final User Experience.

The end result of the User Interaction and Accessible Design of the proposed AI Augmented Food Analyzer application will allow the delivery of advanced A.I. Powered Food Recognition through an interface that is easy to use, intuitive and fully functional for users with visual impairments. The application accomplishes this by providing Users with a minimal step Workflow, Structured Text to Speech Feedback, Tactile Friendly Controls, Native Android Accessibility Service Compatibility, Clear Communication

of Interaction States, Accessible Error Handling Ability, Consistent Navigation Patterns and Multiple Modes of Interaction, therefore creating a meaningful assistive experience for Users by transforming the back end Artificial Intelligence into a real world application which will help Users use Intelligent Technology to not just identify foods, but give them control and independence through usable, accessible, and functional knowledge of their Nutrition.

5.3.7 Performance and Optimization

The importance of performance optimisation became apparent during the course of the design phase of the AI Augmented Food Analyzer application; the practical utility of any assistive intelligent application is based not only on accuracy of recognition, but also on how quickly it responds, how efficiently it performs and how smoothly it interacts with its users in real time. Even if the accuracy rate of an AI system is high, it may not be practical to use in the real world if the time for it to infer is too long, the application does not respond in a timely manner or it consumes too many resources thus creating operational instability. This is especially important in assistive mobile applications where users expect immediate and reliable responses from the application during daily use. Accordingly, performance optimisation was consistently treated as a major engineering objective throughout the design process and back end implementation of the application, with efforts directed at optimising the entire end to end pipeline as opposed to optimising each component in isolation.

A major contributor towards performance optimising the application was the decision to implement a server side inference architecture, which offloaded the computationally intensive deep learning processes from the mobile device to a back end server. This decision on architecture greatly

reduce the computational load on the mobile device, optimise memory utilization on the mobile device, reduce battery drain and increase the responsiveness of the application. The mobile application was designed primarily to be a thin user interface client communicating with the back end server; by doing so, the system is not subject to the latency and resource constraints that would otherwise occur if a large scale neural network were run directly on a mobile device.

Within the backend inference environment, optimization began with model loading strategy improvements. Rather than loading trained deep learning models into memory for each incoming request, all deployed models are initialized and preloaded during server startup. This persistent in memory model hosting eliminates repeated disk loading overhead and significantly reduces per request inference latency. Model preloading is a standard optimization technique in production AI deployment because repeated model initialization can contribute substantial avoidable latency during inference serving operations.

The image preprocessing pipeline was also optimized to reduce unnecessary computational overhead. On the client side, captured images are resized and compressed before upload to reduce network payload size and transmission time. On the server side, preprocessing operations were streamlined to include only transformations strictly necessary for compatibility with the deployed models. Redundant or experimentally useful but deployment unnecessary preprocessing steps were removed from the production inference pipeline to improve overall throughput.

Backend inference routing was optimized to ensure that only required models and processing branches are executed for each request. The system avoids unnecessary invocation of multiple recognition pipelines when a single appropriate inference path is sufficient for the input scenario. This

selective routing reduces redundant computation and improves average inference time. Invalid or low quality user inputs are prevented from causing any unnecessary processing during the downstream of an API these types of inputs to have excessive processing overhead at the backend of the application, is accomplished using both confidence thresholds and lightweight validation logic.

Efficiency of API communication between the backend and the Android Client. Overall, latency for an API Call is defined as two components: time spent doing computation and time spent in transit between the two between client and server. Optimization of the request response pipeline was accomplished in several ways to reduce any delays in transmission. As much as possible, image encoding, to include structured payloads HTTP formatted and light weight JSON serialization, were utilized to minimize data transfer overhead, while ensuring the image fidelity necessary for recognition.

The network request and image preprocessing tasks were completed on a background thread so as not to block the user interface main thread while waiting for a response from the backend API, this process of performing tasks in asynchronous fashion is a best practice for mobile software performance engineering. The result of this work is increased perceived responsiveness to the user while inference takes place.

When applying optimization techniques to applications, memory management issues were also accounted for. Large bitmap objects with an excessive amount of duplication created in memory, known as "large bitmap objects", can cause memory pressure or instability on a resource constrained mobile device; thus, the way you manage your captured images was implemented to prevent such unnecessary duplication in memory.

To help keep memory usage efficient during the operation of the ap-

plication, temporary image resources are released right after they are processed.

Performance optimization was also applied to the efficiency of database query execution within the backend nutritional mapping subsystem. Since food metadata is retrieved for each successful prediction, databases are set up to allow very fast indexed access with the least possible amount of overhead when performing database lookups. This means the contribution of the nutritional mapping system to the total latency of the system is insignificant and has no impact on the time to infer and communicate the nutritional value of the food.

User testing and profiling of the entire operational pipeline capturing images, uploading images, inferencing, retrieving information from the database, serializing the review information and rendering the results to the user were important for identifying any performance bottlenecks throughout the entire operation. Thus, if optimization of a particular component was implemented without the benefit of end to end profiling, the opportunity to solve the true bottleneck would have been lost. Cumulative delays throughout the total operation pipeline of an AI based distributed system create user perceived latency, therefore, optimizing the value of the inference portion alone will not result in acceptable user perceived latency; end to end profiling is critical for deriving user perceived latency in these types of systems.

A critical design idea used during optimization was the need to consider performance versus recognition quality as balanced elements when making optimization decisions. To ensure that model accuracy and overall accessibility quality did not suffer, optimization techniques that imposed a high risk of de optimizing the model were avoided. Optimization decisions were based on maximizing recognition reliability while minimizing redun-

dant calculation or architectural efficiency. By balancing these two aspects of the system, improvements in responsiveness do not negatively impact the overall assistance provided by the system.

The modular design of the system will also provide additional opportunities for performance improvements in the future, such as through model quantization, hardware acceleration, inference batching, CDN Content Delivery Network backed deployment scaling, or migrating the application to a hybrid edge or cloud architecture if there is an intention to deploy the application widely. By keeping the inference and communication layers modular, the current design has the ability to adopt more advanced optimizations through future iterations of the application.

In summary, the AI Augmented Food Analyzer project was successfully able toimplement performance optimization such that the system has achieved the performance responsiveness and efficiency needed for an assistive application in practice. By using server side inference, pre loaded model storage, optimised pre preprocessing, selective inference routing, efficient API communication, asynchronous processing on mobile platform, efficient handling of memory, rapid retrieval of data from database, and refinement based on end to end profiling; the project was able to accomplish an overall version controlled system with a balanced deployment that is able to provide the user with an accurate identification of the food item in real time. Therefore, through the optimisation of systems capabilities; the project has assisted in making the AI Augmented Food Analyzer System from a technically functional prototype to having sufficient capabilities for an actual intelligent system utilised in the assistance of individuals in the real world.

5.3.8 Application Access Security

Security of applications is extremely important in modern artificial intelligent programs just as much as they are in computer networks. Users share visual input photos over networks; therefore, security of the application's access point is critical for all users that will be using this application once released, even though this application isn't a transactional platform e.g., financial, medical but still processes user generated data and sends to a back end server. Additionally, security of application access is foundational to achieve the trustworthiness of the entire system since a user of a system has to have confidence in how well an application operates, providing reliable & safe services while integrating into their daily lives.

In the architecture of the proposed system, client server communications are controlled in a manner that ensures that the Android application communicates with the backend server only through the API endpoints defined and controlled by the architecture. In this way, the backend server accepts explicitly defined requests and has no access to any of the other functionality within the backend server. All requests sent to the backend server are processed through user defined inference APIs that were built specifically for submitting food images for upload and retrieving food prediction responses. The controlled endpoint architecture limits the attack surface of the backend server and ensures that client server interaction is limited to appropriate operational flows.

The architecture of the system will also utilize encrypted transmission protocols for all the data transmitted over its network. This includes both the images uploaded to the backend server and the prediction responses returned from the backend inference server to the Android client. The use of encryption for all data transferred over the network protects the data

from being intercepted or improperly inspected while in transit, as well as protecting the integrity of the data being sent between the Android client and the backend server. Although there is no highly sensitive information explicitly processed by the application i.e., only food images, secure communication is still a recognized best practice for all mobile applications developed to use network connections.

At the backend server, request verification and structured input management help mitigate the effects of poorly formatted or potentially harmful client requests on inference service disruption due to entering the Inference Pipeline without being accounted for by the system. The validation layer of the system will verify that uploaded requests were presented in the appropriate format and expected to meet a specified content type and specified structural format. By validating what is consumed by backend models in terms of appropriately structured image formats, it is ensured that payloads that do not meet the established criteria will not be supported by the system, which means that they will not cause the Inference Engine to fail or to create instability in the backend server.

Security and reliability of the operation are enhanced by using resource control methods and by applying various types of inference handling safeguards at the backend. Because deep learning inference requires a significant amount of computational effort to process, the potential exists for server resources to be overwhelmed by excessive requests when not subjected to a control system. For this reason, inference requests must be submitted to the backend through a control system that enables orderly processing so that the desired level of system response is maintained when the system is functioning under standard operational conditions. In production, controls on inference requests may be further built upon by using other methods such as rate limiting, load balancing, and queue manage-

ment to improve organizational resilience under high demand conditions.

Model protection and backend isolation are also crucial factors for security. The trained deep learning models are hosted only on the backend server and not directly accessible to the client application. Because of this design, users and third parties cannot extract the model's weights or reverse engineer the proprietary configuration of the model, and they cannot manipulate or tamper with the recognition logic deployed by the client application through client side manipulation. Additionally, utilizing a centralized approach to the hosting of the model not only simplifies ongoing maintenance but also gives the model provider more control over inference assets and provides stronger protection of intellectual property associated with the model.

The Android application design incorporates secure access practices for the client side by limiting how much of its internal logic is exposed to the client and by controlling the user interaction strictly through the defined workflows in the client application. Internal backend URLs, request structures, and inference processing logic are abstracted out of the user interface so as to limit the potential for improper usage and diminish the potential attack surfaces associated with direct manipulation of the client. Academic prototype applications may not have complete production hardening, but the architecture of the application provides for the potential for future integration of stronger client side protection mechanisms as needed.

The ways to deal with error situations in security design were handled with care and consideration. The application only provides controlled friendly error messages to give to users when communication with the server does not occur, or the inference processing does not occur, eliminating the possibility of leaking backend implementation details while providing feedback for the user to understand. It is common for secure design to

lack transparency about the details of internal architecture to users that are external to the organization.

In addition to providing technical controls; the architecture of this system supports centralized maintenance and monitoring of the backend application. This centralized system provides support to enable the rapid release of model updates, quick application of server patches, and controlled release of enhanced inference pipelines without requiring modifications to the client application. Since the inference logic is located in the backend application, all users can be updated immediately to correct vulnerabilities or bugs, thus, preserving the maintainability of the system and decreasing the overhead of managing security.

From a practical standpoint, the security requirements for this project were designed with consideration for both the academic scope and proposed deployment context of the project. Although the level of security required for enterprise scale access authentication, multi user authorization, advanced intrusion detection, and distributed infrastructure hardening are not required or expected in the current system, the architectural techniques established provide a direct path for future development of enhanced production level security methods if when broader deployment is pursued.

The security measures included in the design for the AI Augmented Food Analysis application focus on protecting client server communications, controlling access to the back end of the system, validating requests from the Application for Inference Services (AIS), isolating trained models on the back end server, preserving the reliability of the Operations of the Application, and preventing any unnecessary exposure of internal back end components to clients. All these measures will ensure that the Intelligent Recognition Service can be maintained securely while providing

the end user with a smooth user experience. Integrating the security considerations into the application architecture during implementation versus treating them as afterthoughts aligns this project with current software engineering principles as well as modern practices in deploying Intelligent Systems.

5.4 Database Security

The proposed AI Augmented Food Analyzer's database security is a key part of ensuring the overall success and trustworthiness of the application, as the database will serve as the primary storage area for the food metadata. Although the database will not have highly sensitive personal data, protecting the food data stored in the database for integrity, availability and consistency is important because incorrect,invalid corrupt data stored in the database could negatively affect the nutritional information and data that users will receive as a result of using this application. Because assistive applications often rely on users relying on the system's outputs for making sound choices, low risk data must be adequately protected.

A backend isolated database access architecture is the main security strategy for the database layer. The database is only available on the backend server, so there is no direct access from the Android client application to the database. All client access to food metadata is indirectly through the backend inference API, which sits between the mobile application and the database. This architecture prevents an external user from directly querying, modifying, or inspecting the contents of the database and dramatically reduces the exposed attack surface of the database system. Reducing the direct exposure of the database is a key principle of secure application architecture and is important for maintaining both data integrity and schema

confidentiality.

The logical control to the database is managed programmatically through backend server logic; thus, only authorized backend processes can perform read functions needed for retrieving nutrition information. By means of having all database interactions occur within the backend application layer, the system can maintain strict query execution control and eliminate channels for arbitrary client generated database commands. A layered access model increases security, while at the same time streamlining maintainability and validity of database processes.

In order to maintain data integrity, the database schema was designed to have standardised and validated food entries, so that there is consistency in respect of food labels, calories and ingredients' description across all record entries. By validating and normalising data prior to insertion into the database, the occurrence of malformed records, duplicate records, or inconsistencies in the database can be minimised. The importance of data consistency in this application is magnified, because if recognition labels do not match with stored food entries, the application may return an incorrect amount of nutrition or fail to allow a lookup operation.

The security of the database is also maintained by preventing the unauthorised modification of the data located in the database because all food metadata exists in the back end of the application and cannot be changed by end users via the client application. To allow for the administrative modification of records in the database, the administrator must have access to the backend server environment; therefore, editing privileges are limited to those who are authorised to maintain the systems and cannot modify nutritional values, food mappings or ingredient data through the front end.

This method also includes some form of controlled access for executing

queries to ensure that all operations on the database are done through a predefined method lookup routine to allow for structured based on labels provided through the pipeline retrieval of records versus unrestricted or dynamic creation of queries at time of request. With this limited scope for querying, the amount of interaction with the database will be reduced as well as the potential for unintentional or improperly structured query activity. The implementation of controlled queries for developing applications that utilize secure database systems is beneficial through reducing attack vectors and providing improved predictability of system behavior.

When it comes to design for the database, its main focus will be to adhere to providing an inexpensive means for accessing the data between user and database system easy to find the item versus updating records no use of these records for transaction processing. Because the database can only be used in providing data i.e., it supports read only access, designing for providing high levels of availability and reliability must also be considered as a matter of building data confidentiality. Specifically, failure of the nutritional lookup would result in an incomplete i.e., unable to provide the requested information recognition of an item even when the visual inferencing would be able to function. As such, the database architecture has been implemented such that it is located within the backend with the intent to provide stable performance for querying as well as providing where you can retrieve a query if one is being requested by any user requesting retrieval of a query through the visual recognition process.

Centralized maintenance and update control offer the database layer an architectural advantage. With food metadata stored server side only once, the ability to make changes in calorie counts, ingredient description, food labeling, expanded nutrition records can occur centrally without changing the Android client application. This will ensure that all users have access

to the latest food metadata immediately upon deployment of back end updates. Centralized maintenance also enhances the long term security management of the database by limiting the number of distributed copies of data that need to be synchronized or protected.

Backup and recovery are other considerations that support the reliability of the database. While the project is currently in prototype form and operating within an academic context, standard databases' backup strategies are designed to protect food metadata against accidental deletion, corruption, or backend failure. In practical implementation scenarios, regular backup procedures would be critical for achieving system continuity and clarifying how to protect operational data.

From a privacy perspective, the database design does not save unneeded personal user information. The primary focus of the application's food recognition and nutrition assistance is structured food metadata the data needed to produce a recognition result. Reducing the amount of saved personal data within the system reduces the risk to privacy and makes it simpler to manage data.

Security measures have been employed in the database design of the proposed AI Augmented Food Analyzer such that the nutritional and associated food metadata will remain secure, reliable and consistent in use, both in storage and throughout functional use. The many layers of the 'database layer' include a backends to isolating access architecture, controlling how queries are handled, not allowing an end user to modify information, having a centralized method to maintain the information stored in the 'database layer' and ensuring that all information is validated when stored to guarantee data integrity. All of these layers assist in ensuring an accurate transformation from the output of AI recognition to real, meaningful dietary data for the user of the application. This can help ensure

accurate nutritional intelligence through this user's use of the application and also provides a level of confidence in the integrity, reliability and protection from unwanted outside influences of the overall assistive system.

Chapter 6

System Testing and Evaluation

The final stage in the creation of the AI Augmented Food Analyzer involves testing and evaluating the system's performance. System testing evaluation is the most important stage of the entire AI Augmented Food Analyzer series of development. The previous chapters provided detailed descriptions of system design, implementation and application development, and this chapter focuses on determining how well the system performs in a real world environment. The second stage of evaluating the system is determining how accurately it performs, while the other stages that need to be assessed include reliability, efficiency, usability, and robustness.

As a result, because the AI Augmented Food Analyzer is designed for use as an assistive application for visually impaired people, the scope of testing will not only evaluate the performance of the system, but also the level of consistency in its results, the amount of time it takes to deliver information to the user, and the type of interaction that is provided to the user through the auditory output. Thus, quantitative and qualitative methods of evaluation will be employed to evaluate whether the AI Augmented Food Analyzer has achieved its intended purpose.

6.1 Testing Strategy

To ensure the proposed AI Augmented Food Analyzer functions correctly in the real world, we developed a comprehensive and layered evaluation framework. We aimed to validate both the technical accuracy of the developed software and the reliability, robustness and readiness of the complete intelligent recognition system. Traditional software systems often use deterministic logic, allowing them to be verified with simple, functional tests; however, AI driven applications differ in that their underlying machine learning models create probabilistic outputs that depend greatly on

data quality, environmental context and real world variability. Therefore, this project's adopted testing strategy goes well beyond typical software verification by incorporating both structured machine learning validation and integrated systems testing, along with assessment methodologies that are deployment oriented and focus on evaluating the usability of the entire system as a whole, to verify that the AI Augmented Food Analyzer will work consistently under normal operating conditions.

The machine learning component of this testing process began with dataset based validation and evaluation of models used to recognize foods. These models were created using different sets of the food image recognition dataset for both training and testing of the models so that the test datasets were completely different held out from the training datasets. The model's performance assessment on the held out datasets could then be used to accurately determine how well the model was generalizing because it would not be memorizing the training examples.

Since evaluation of a model's performance on random unseen images is critical for confirming that they will operate as expected when exposed to new foods during actual use, performance metrics were reserved for every phase of the model development to stay as true as possible when estimating the performance of the model.

To provide a quantitative measure of the quality of recognition, the testing strategy was developed to include various task appropriate metrics associated with specific evaluation criteria for each task classification or detection. The AI system is capable of performing two main functions food classification and food detection and therefore, various metrics were included as part of the evaluation process to provide insights into the model performance, performance overall correctness, and detection errors including number of false positives and number of false negatives and localize

errors in detecting a food item when appropriate. These metrics also allowed for a more complete understanding of the model performance than just the percentage of samples classified correctly and provided a mechanism to optimize the models through data driven refinement. The training objective of the model is defined using cross entropy loss as shown in Equation 6.1.

$$L = -\sum y_i \log(p_i) \tag{6.1}$$

where:

- L is loss
- y_i is true label
- p_i is predicted probability

The project was developed by utilizing an iterative process of experimental validation during the model development phase. For this purpose, several iterations of experimental comparisons were conducted on multiple model design attributes, augmentation strategies and methods for optimizing models algorithm performance. By conducting comparative evaluations of different deep learning architectures and preprocessing configurations under standardized conditions, we were able to identify which of the alternatives would be the best deployment candidates. Thus, this iterative experimentation provided us with data to guide our architectural refinements so that our model selection and optimization decisions were based solely on empirical data rather than theoretical preferences.

At the engineering level, our testing strategy utilized a layered validation process that included unit tests, integration tests, system tests and acceptance tests. Each software module, including image preprocessing routines, API communication handlers, database retrieval functions,

and text to speech components, were independently tested to confirm they functioned properly alone and then together, through integrations tests performed after they were validated individually to test how the modules and components interacted and communicated with each other and to ensure the accurate exchange of data after module and component boundary crossings. Layered testing has proven to improve overall reliability because it provides feedback earlier in the development process, especially with respect to testing the component itself, thus reducing bug propagation and eliminating the need to backtrack through large systems later in the overall testing process.

Once all subsystems were interconnected to create a full system, end to end testing was performed to test the entire operational pipeline of each image capture down through backend inference, nutritional data retrieval and vocal output delivery. This full system testing ensured that the application integrated all modules so they worked together as a single user interface rather than simply as a group of independent modules. In distributed AI applications such as this, it is extremely important to perform end to end testing because failures in these types of applications may happen through the normal interactions between the various integrated subsystems, while each of the individual modules may function perfectly by themselves.

Because this application is intended to be a usable assistive technology, the testing also took into consideration the actual conditions at the time the technology will be used. Rather than limit testing to ideal or laboratory conditions, the system was tested with different images of food that had varying amounts of illumination, angle of photography, style of presentation, type of table or plate used to photograph the food, complexity of the background and types of food photographed overall. These realistic condition tests allowed for an accurate assessment of the robustness of the

software when used in real life situations such as photographing food in restaurants, kitchens, cafeterias and outside in non ideal environments. If the only evaluations done are using calibrated test cases, then it is likely that the system is over assessed considering whether or not it is ready for the real world.

Additionally, mobile application testing was performed while taking into account different device types and networks, as practical operation depends on not just the quality of the model used for AI but also on how well the app functions from various Android devices and the different types of connection conditions that exist. The mobile client was tested with different hardware configurations, screen sizes, camera capabilities and network conditions to verify that the app behaved consistently when deployed in real world mobile environments. The broader scope of testing shows that AI driven mobile applications should be validated as complete distributed systems and not just as independent machine learning models.

Accessibility testing was also done to ensure that the app can be used practically by people who are visually impaired by validating how output from text to speech is sequenced, verifying if the app works with the screen reader's vocal commands, how well people can touch the interface, how users can navigate through the app without losing their place, and how all elements of the app's workflow are consistent with every other element. Since individuals who are visually impaired typically rely on assistive means of interacting with the system, validating accessibility is critical when performing practical testing on the system as opposed to being just an added touch of appearance.

Another critical dimension of testing methodology was the evaluation of error and fault tolerance. The application underwent controlled testing under unnatural and unique situations, which included testing with

invalid images, testing with intermittent network access or failure, evaluating predictions that were low in confidence, considering backend failures, and designing requests with malformed structures or formats. By doing so, it was ensured that the system would respond in a controlled manner to error conditions and would therefore be less likely to experience instability or crashes. Reliability and consistently high levels of performance from an assistive technology application have a major impact on how much user trust is developed and therefore contribute to ongoing usage.

The testing process was implemented as an iterative rather than exclusive end phase of testing to collect therefrom experienced based knowledge through the entire test environment and application development life cycle. The results of each phase of the testing effort were input into modifications to the models and overall architecture, as well as adjustments to interface design operation, backend logic, preprocessing methodology, and production implementation. Thus it may be concluded that this iterative feedback driven approach to testing in conjunction with the various tests completed by each testing cycle allowed for continuous improvement in the overall application product as well as the method of production and deployment of that products.

All in all, the testing approach utilized for the AI Augmented Food Analyzer being developed as an overall validation framework which brings together machine learning validation against many factors including Machine Learning performance, Software Engineering Verification, Integrated Systems Testing, Assessment of Performance in Realistic Deployment Conditions, Assessment of Validity of Accessibility and Fault Tolerance Assessment. The combination of quantitative AI Performance Testing against the practical application testing to the design user application distribution plan yielded a comprehensive validation strategy to ensure that the

final solution was evaluated for more than just a technology level and ultimately demonstrated true operational readiness for completion of realistic assistive usage scenarios. This best practice provides assurance that the solution can produce results that can be consistently, accurately and reliably produced when used with practical usage case scenarios.

6.2 Evaluation Metrics

A simple assessment of the correctness of the predictions alone does not suffice for evaluating an AI based assistive recognition system. The AI Augmented Food Analyzer uses multiple types of technology that includes image classification, object detection, backend inferencing and assistive interaction. As such, to evaluate the AI Augmented Food Analyzer, it will be assessed using a variety of evaluation methods to evaluate performance on reliability, responsiveness, and practical usability, and predictive accuracy. Evaluating an intelligent assistive system solely based on an overall accuracy measure could lead to an incomplete or inaccurate picture of performance in real world environments; therefore, a multi dimensional evaluation framework will be used to more fully evaluate the AI Augmented Food Analyzer across three dimensions: recognition quality, operational efficiency, and user facing responsiveness.

Classification accuracy is defined as the ratio of correctly classified food categories to the total number of food categories evaluated, providing an overall estimate of performance and commonly being the baseline performance metric for image classification research. Although classification accuracy provides a broad picture of the model's performance, it does not reflect the distribution of errors in classification or whether classification performance is balanced across categories. In addition, in food recogni-

tion tasks where there may be visual similarity between certain types of food or class imbalance, classification accuracy alone will not provide a complete evaluation of performance.The classification accuracy metric is defined mathematically as shown in the equation 6.2.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{6.2}$$

where:

- TP true positive
- TN true negative
- FP false positive
- FN false negative

To improve on this shortcoming, precision and recall have been added to the evaluation framework as additional metrics to provide greater depth in the understanding of the ways the model predicts. Precision assesses how many of the predicted positives were actually correct, hence indicating how well the model avoids producing false positives. The importance of high precision when predicting food items is that when the system predicts that a particular food item was found, there is a very good chance that the prediction will be correct. Whereas recall assesses how well the model detects positive samples the item exists from those that are actually in the data set, hence measuring how well the model has detected relevant food categories without omission. Thus, high recall is very important for assistive technology, since missing a food item from the present inventory can significantly affect the user's trust and usefulness of the system.Precision is calculated

as shown in Equation 6.3.Recall is computed as shown in Equation 6.4.

$$Precision = \frac{TP}{TP + FP} \tag{6.3}$$

$$Recall = \frac{TP}{TP + FN} \tag{6.4}$$

Precision and recall have been known to have a trade off, which is why it was important to use an aggregate evaluation metric such as the F1 Score. The F1 Score is the harmonic mean or average of precision and recall, representing a summary of a prediction's performance based on both types of errors and their relative importance as the two types of errors can be equally detrimental to the prediction's quality. Thus, having a balanced approach to measuring the prediction's quality through use of the F1 Score is especially valuable for food detection systems because of the importance of balanced reliability within the detection system to provide accurate predictions over the total quantity of food items within a given period of time.The harmonic mean of precision and recall can be calculated according to the following formula 6.5.

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{6.5}$$

The evaluation of the YOLOV8 food detection system also involved calculating the Intersection over Union (IoU) and the mean Average Precision (mAP). The IOU is used to measure the overlap between predicted locations for detected items bounding boxes with IRM or actual object locations using object detection's IRM system to generate predicted object locations in the bounding box when there is overlap between these two measurements, thereby accurately describing the localization quality

or accuracy of predicted food items. The Mean Average Precision metric takes the concept of object localization accuracy one step further by providing an evaluation of the localization and classification accuracies at various confidence thresholds and for each item class to provide a complete summary of the accuracy both the classification accuracy and localization accuracy of detected food items based on multiple classifications at various confidence thresholds.

In addition to evaluating who made each prediction accurately, the evaluation framework also included confidence score analysis because it is a primary measure of how reliable that inference will be. A deep learning model provides a confidence estimate with every prediction it makes. This confidence estimate is how certain the model is regarding its classification or detection result. Analyzing these confidence values can be used to establish metrics for determining how reliable each prediction will be and also provide metrics for determining which low confidence outputs should be filtered e.g., not processed. The use of a confidence based analysis approach is particularly valuable in assistive applications, where uncertain predictions should be either omitted or flagged to maintain the user's trust in the system.The proposed system uses the following rule to make decisions as defined in the equation below 6.6.

$$Prediction = \begin{cases} Accepted, & \text{if Confidence} \geq 0.85 \\ Rejected, & \text{otherwise} \end{cases} \tag{6.6}$$

where:

- $Confidence$ is model prediction score
- 0.85 is threshold used in system

The operational performance of the deployed solution was assessed us-

ing inference latency metrics. The definition of these metrics included assessing the entire time elapsed from the moment an input image was captured, to the moment a usable recognition response was provided back to the user by the application. The proposed application utilizes a cluster of client server computers that work together as a distributed architecture, therefore, when measuring latency we must include all components of the end to end pipeline e.g., image capture, client side preprocessing, network upload, server side inference, database retrieval, serializing the response, and rendering at the client side in the latency evaluation. End to end latency is critical in mobile assistive applications because when latency is very high, the user will not be able to use the application effectively, giving the user a perception of unreliability.

In addition to assessing the responsiveness of the mobile app, a variety of user interface performance metrics were evaluated, including; Smoothness of screen transitions; behaviour during loading; stability of asynchronous processing; responsiveness to interactions when another network bound operation is occurring. While not as formally defined as those for assessing AI, these metrics, collectively, will substantially impact the overall quality of the application as it is delivered in practice and ultimately affect user satisfaction.

Furthermore, accessibility related evaluation was incorporated into the overall metrics framework. Although quantitative measurement for accessibility may not always be possible, the qualitative criteria used to evaluate speech feedback clarity, touch control usability, screen reader compatibility, workflow simplicity and assistive effectiveness are essential; As the success of an application is determined not only by technical accuracy of recognition but by the practical use by visually impaired individuals.

To ensure that the interpretation of results was robust and meaningful,

the evaluation metrics were assessed as a whole, not alone. For example, a model may score well in both accuracy and recall; yet certain food items may not be correctly included in the model due to its high accuracy with low recall. Likewise, low latency will have little value if the degree of recognition quality is not upheld. In this way, by producing results that demonstrate complementary strengths and weaknesses, the project was able to ensure that optimisation was performed based on a holistic view of system performance rather than based solely on mitigating overfitting from any one evaluation metric.

A different use of evaluation metrics also supported more informed development through a more iterative approach. By conducting analyses on the various evaluation metrics during the experiments, we were able to detect areas of improvement within the respective models, preprocessing pipelines, and deployment configurations. In turn, this allowed us to accurately guide the improvement of our system throughout the development stage. As a result, the final deployed system exhibited improved predictive performance, as well as enhanced operational usability.

As a whole, the evaluation metrics utilized for the proposed AI Enhanced Food Analyzer created an inclusive framework for evaluating how the system performed in terms of recognition quality, localization precision, confidence reliability, operational responsiveness and practical usability. The aforementioned metrics combined to create a multi dimensional approach to understanding system effectiveness through the utilization of classification accuracy, precision, recall, F1 Score, IoU, mAP and confidence analyses and measures of latency and accessibility. Hence, the proposed system will be evaluated as not only a machine learning model; rather, it will be evaluated as a complete assistive intelligent application that operates in real world situations.

6.3 Experimental Results

The evaluation of the AI Augmented Food Analyzer yielded experimental results that demonstrate the effectiveness of both the recognition system and its use as a tool to assist users with visual impairments. The experimental results provide both quantitative and qualitative information about the performance of the AI Augmented Food Analyzer. Additionally, the results illustrate how the project progressed from a model development and training perspective to testing the final product under both controlled and real world situations. The experimental results provide insight into the efficacy of the model selected for use in the AI Augmented Food Analyzer. Several model architectures, such as ResNet101, EfficientNetB4, and ConvNeXtSmall, were evaluated using standard test datasets. The analysis of the results showed that there is a statistically significant difference between the models based on their performance, thus highlighting the differences between model architecture depth, the amount of features being used to identify food, the computational power needed to run these models in real time, and the suitability of the models to be used in conjunction with the AI Augmented Food Analyzer.

While most of the models were able to identify and classify food items as expected, some architectures had stronger discriminating capabilities than others for certain types of foods. For instance, while deeper models produced stronger discriminating capabilities for identifying food items in complex food categories e.g., food with very little texture and very similar colors, lightweight architectures performed faster for identifying food items due to lower computational complexity but at a cost of producing lower predictive performance.

The classified models evaluated showed the best accuracy at 80%+

on the evaluation dataset, which indicates a good level of recognition for the different types of food. The accuracy level demonstrates the finalized model was able to learn a general visual representation for distinguishing between food classes despite issues associated with food recognition, including significant variability high intra class difference and considerable visual similarity between classes. To achieve an accurate model required using an appropriate choice of model and effective use of augmentation, preprocessing, transfer learning, and optimization during all stages of development.

The experiment also showed that saliency based preprocessing and attention to visual refinement improved prediction stability in visually cluttered scenes. The ability to focus on items, and remove nonfood items in the images, improved the overall robustness of the system when evaluating food images with visually distracting background items such as utensils, patterned table surfaces, or complex plates. These results support the decision to include saliency based preprocessing in the broader food recognition pipeline.

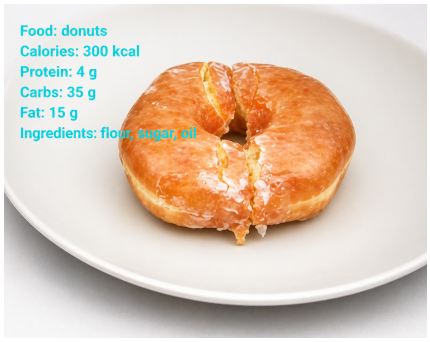
In a food detection use case, the integrated YOLOv8 detection pipeline showed good object localization and identification across multiple food types within a single image frame. The results from experimentation indicated that the detection model showed accurate detection and localization of visible food areas belonging to a multi class category in a standard presentation condition for food. Thus, it has been able to expand beyond very simplistic, single label classifications of food and move into more realistic, multi label dining scenarios. The detection resulting from the framework has been evaluated through measurement of the localized overlap of the detected objects from the correct position and an analysis of confidence in the detected position, which supports the assertion that the integrated

detection framework yields useful object detection capability with a degree of reliability suitable for practical usage.

Beyond detection accuracy, the experimental assessment of the inference performance showed that the client server architecture used allows for near real time operational response time, making it very suitable for mobile assisted use. The time to perform end to end inference from image capture to upload, server processing, database access, and response delivery were well within acceptable thresholds for use during the testing that was done on the system. There will be variability in response times based on network conditions and server load, but the optimized construction of the deployed pipeline generated responses quickly enough to maintain an uninterrupted user interaction based on expected performance.As shown in Fig. 6.1,The proposed classification model has also been tested against a variety of food products, showing its ability to classify accurately food items from separate categories and provide the associated nutrients found in each food item, as illustrated in the figure..

(a) Samosa

(b) Spring Roll



(c) Donut

Figure 6.1: Testing results of the proposed food classification and detection model on different food samples

The system showed that its capability to learn was not limited to just how it learned from idealized data, since it was tested under many different types of real environments where food is routinely prepared and served e.g., different lighting conditions, angles of capture, background images, and type of meal. Although the majority of test results related to food image recognition performed well on moderate changes, there was still less confidence in predictions and lower accuracy for food images that were very difficult to recognize because of extreme lighting too bright or too dark, significant obstruction, very unusual angle of capture, or extremely cluttered background. These findings support previous studies that have

addressed the boundaries of computer vision systems recognizing objects in real world environments without constraints placed on them by artificial limitations imposed by controlled settings.

Testing at the application level demonstrated that the mobile application, backend server, database retrieval layer, and accessibility mechanisms had been successfully integrated into an end to end functional operational platform. Food Images taken on the Android client device were successfully transmitted to the backend server, processed through the deployed artificial intelligence pipeline, augmented with food nutritional metadata, and returned to the Android client application for visual spoken feedback presentation. Therefore, the system functions not only as a separate machine learning model but as a fully integrated assistive application.

User centered testing confirmed that the program's accessibility features text to speech feedback and easier workflows for interaction enabled all users to successfully operate this application using nonvisual methods. Together, the speed, clarity, and simplicity of the spoken feedback and the workflow all produced an effective interaction model for assistive technology users. No formal user studies were conducted in a clinical setting as an objective of this project, but functional testing of accessibility demonstrated the application's effective design and alignment with its intended use.

The results from these experiments also revealed some limitations that will inform future development of the product. Some food category items that visually appeared to be similar to each other i.e., overlapping color distributions, similar looking textures, and or presenting similarly were often misclassified. Also, when two or more food items were mostly overlapping or in small areas of a camera image, the ability to detect those items decreased significantly. These findings provide evidence of the ongo-

ing challenges of performing fine grain recognition tasks with food items and indicate the need for additional datasets and algorithms to improve future versions of the product.

The experiment results show that the AI Augmented Food Analyzer achieves its primary design criteria of properly identifying foods accurately while being able to be used on a mobile device with easy to use interfaces. The results from comparative testing show that the system has performed well in terms of it being able to connect model performance to the overall classification performance of the created models greater than 80%, provide accurate food detection, have a near real time or some cases actual real time inference response to food item identification and integration into the whole system. Thus, it is possible to utilize deep learning in combination with a mobile application to provide a practical means to help perform food analyses in an assistive manner. The experimental results support the fact that the proposed assistive food identification system is technically feasible and has significant promise as an intelligent food identification system with accessibility as a primary guiding principle.

6.4 Application Level Testing

Application testing was performed to validate the full mobile software product operation of the proposed AI Augmented food analyzer. Even though testing of the underlying deep learning models demonstrated recognition capability under controlled testing conditions; testing of the entire application workflow will validate that the integrated Android application with the supporting back end infrastructure will relate together as a stable, responsive, and ready to use assistive platform in a real operational environment. The complete application workflow includes mobile

interface functionality, camera integration, back end communication with image transfer and handling of the server's response, rendering of nutritional data, generating speech output, and fault handling.

The functional verification portion of the testing phase was done with the main workflows of the application being performed. Each of the user interactions supported by the primary users was methodically monitored during the testing to confirm that they performed the intended action, from initial launching of the application, through to recognition jobs being completed including camera initialized correctly, image captured correctly, image uploaded to the back end service correctly, handling of a server response correctly, rendered nutritional data correctly, and generated text to speech accurately. Testing confirmed that the full recognition process of the application operated correctly without requiring user intervention or performing actions that were not supported; thus validating the integrity of the end to end operational pipeline.

Camera capture subsystem had a primary focus during testing due to its crucial role in using image capture for the beginning of the Recognition was an integral step in determining overall performance of the whole system. Consistent testing of image capture was also performed across various Android devices and different hardware configurations for things such as how well preview renders to display, stability when capturing an image, how correctly the camera handles orientation changes and how well it monitors permission grants. Additionally, the tests confirmed reliable operation of the camera subsystem across all supported devices, and produced images that were suitable for downstream inference processing.

The Networking and Backend Integration Layer were extensively tested to ensure that there was reliable client server communication under real world operating conditions. Testing of the uploaded images was performed

and verified successful upload to the backend Inference Server and proper parsing and display of the returned prediction payload in the client application. Additionally, tests confirmed that the application operated stably under normal network conditions, and properly managed the varying response time without impacting the user interface. As the application depends on a distributed backend inference system, it was vital to verify through reliable communication testing that the application can be deployed practically.

The text to speech and audio feedback capabilities of the application were also thoroughly examined to guarantee that recognition results were communicated appropriately after backend processing. Tests proved that recognized food names, calorie values, and ingredient summaries were verbally communicated in the order intended and with accurate pronunciations using the Android text to speech engine. The timing of the spoken output was verified so that the speech would only be initiated following successful receipt and interpretation of the prediction data and that the speech produced maintained logical, synchronized flow of interaction.

To test the application's robustness, it was tested under numerous error and exception conditions, such as: images being captured for various invalid reasons, no images being submitted by a user, the network was lost while the image was uploaded to the back end system, the system could not respond to a user within an appropriate timeframe, API misconfiguration, and the application cannot connect to a server due to unavailability. Each instance was reviewed to evaluate how the application would respond in a graceful manner with an error prevention mechanism that would not cause the application to crash or be in an unstable state. Tests verified that structured error handling mechanisms successfully intercepted the exceptions and communicated appropriate user readable feedback to the

user while maintaining application state stability.

Performance orientated testing of the application was performed to assess latency responsiveness and smoothness of run time during typical operation. Interface lag and loading delay, as well as memory pressure and UI blocking, were monitored for repeated recognition cycles. Based on testing results of asynchronous backend communication and background processing methods ensuring interface responsiveness during the inference processes, the user would continue to be able to use the application even during stages bound by network processing.

Compatibility testing was performed on multiple Android devices using different emulator configurations to evaluate how the application would run based on different hardware and software configurations. Testing was completed by looking at the effects of screen size, resolution, android OS version, memory available, and camera hardware to ensure there was wide operational compatibility of the application throughout the Android ecosystem. By performing compatibility testing, it was verified that the application provided a consistent user interface and functionality across many distinct deployments.

The accessibility implementation was also evaluated as a part of testing. Testing was done to evaluate how the application would work with a screen reader for people who are visually impaired; how tactile navigation flowed; how accessible buttons were; how clearly spoken output was; and the order in which focus was traversed. Based on this accessibility testing, the application's assistive design goals were preserved in the actual deployed software design and did not remain as theoretical design objectives.

Continuous operational testing was performed over several recognition cyclestoassess runtime reliability and stability throughout continuoususe. The testing consisted of running a sequence of image captures, inference

requests, and user feedback sequences in order to identify potential performance degradation, memory leaks, or issues with state management as a result of prolonged operation. The stability of a system with continuous use is especially critical in practical deployments because users may interact with the system several times daily.

Findings from application testing showed that the developed platform is functionaland supports the use of a multi function assistive system. Core workflows completed successfully, backend communication remained stable, inference response processing functions correctly, and accessibility mechanisms worked as intended; furthermore, fault conditions were handled smoothly throughout testing. Minor latency differences due to varying network conditions for real time communication to backend systems have been noted, however there have been no critical failures or indications of instability during standard operation testing that will prevent deployment .

All of the application testing results also confirm, that the proposed AI based model food analyzer is not simply a functional AI model prototype, butcan also be deployed as a functional AI based mobile software application suitable for use in the real world. Application testing validated all functional workflows, communications to and from the backend, camera integration, accessibility mechanisms, runtime stability, compatibility, and fault tolerance. As a result of these findings, the overall application determined the entire integrated application is capable of meeting its intended assistive functionality in both a practical and reliable manner when fully deployed. Therefore, the current testing results support that the developed platform is ready for supporting users intelligent food recognition application.

6.5 Usability and Accessibility Evaluation

The level of success that is achievable by the AIAugmented Food Analyzer is not simply determined by how accurately it can perform its technical functions; it will also depend upon how comfortably, efficiently, and independently a given end user can utilize it in everyday life. In particular, when considering assistive technologies usability and accessibility, design principles such as usability and accessibility must be considered as primary factors when attempting to determine the overall value of any assistive technology system. Regardless of how sophisticated an algorithm may be or how advanced a device's technical capabilities may be, if users find it to be confusing, difficult to navigate, or not accessible, the device will not fulfill its intended purpose and will not assist the user in achieving independence. Therefore, usability and accessibility evaluations formed a significant component of the overall assessment of the system to evaluate whether or not this platform could translate the technical intelligence of the system into actual user empowerment.

With respect to the usability evaluation, the first focus for evaluating the application is the simplicity and intuitive feel of the interaction workflow between the user and the application. The application was designed to operate with the least number of operational steps and the least number of intermediate user input notification actions between capturing an image and receiving recognition feedback. The results of the observational evaluation tests and user navigation tests have indicated that using the least amount of intermediate user input notification steps to complete an operation significantly decreased the overall complexity of operating an assistive mobility technology device and reduced users' cognitive load associated with remembering multi step navigation procedures. Developing

the application to eliminate unnecessary menus, nested choices or configurations from the user's interaction workflow supported a more natural and efficient interaction experience that was user expectable of an assistive mobile technology application.

Another aspect to consider when designing an application is the ease with which users can learn how to use it. Since the intended users may have different amounts of experience with smartphones and technical knowledge, the application needs to be able to teach them its core capabilities with very little initial instruction. The core layout of the application will be the same across all screens; buttons will be placed in predefined locations within each screen; and users will have a linear series of interaction steps that will facilitate the user's ability to learn how to use the application and also minimize the barrier to using it. If a particular application were to require a great deal of training or explanation to use, it would significantly decrease the user's ability to use that application's features in a practical manner on a regular basis.

The primary focus of the accessibility assessment was on the effectiveness of the application's auditory feedback mechanisms since these represent the main means of communication for visually impaired users. The quality of the integrated text to speech output was assessed for clarity, pronunciation accuracy, timing, and sequence of information. Based on the assessment of the quality of the spoken output, users will be able to receive spoken information about the food they are trying to identify e.g., name of food, number of calories, and list of ingredients in an easily understood way and in an orderly fashion e.g., in an easy to follow sequence. The orderly sequencing of information will enable users to naturally process the auditory feedback without feeling overwhelmed or confused by what was provided. In addition, since auditory feedback has been estab-

lished to be used to provide the same information to users that they would otherwise obtain from visually inspecting the output in the intended context, providing effective auditory feedback to the visually impaired user is extraordinarily important.

Investigating the tactile accessibility of the user interface was conducted by evaluating touch target size, spacing, and overall simplicity of layout. In addition, interactive controls were created with very large button areas and sufficient space for accurate touch interactions without visual precision. The results of testing have shown that these design decisions have increased ease of operation during non visual use and decreased accidental activation of unintended controls. Tactile accessibility is especially important for users who are blind since they depend on their sense of touch to explore the user interface of their smartphones and cannot use visual methods to locate controls.

Also, the application has been validated for compatibility with platform level accessibility services like Android Talk Back so that users with visual impairments can interact with the application using familiar assistive smartphone workflows. Screen readers announced each of the interface elements accurately, focus traversed according to a logical progression, and controls were semantically labeled to support meaningful auditory navigation throughout the interface. This type of compatible design will allow users who already use accessibility tools from Android to add this application to their existing assistive device usage patterns without having to learn a completely new way to interact with their assistive technology.

Usability testing took into account how the system provides feedback during different processing states, the application's effectiveness in communicating the status of the processing, and the methods by which the user could receive notice regarding loading of the process through loading in-

dicators and optional audio. Providing feedback while the application was processing data reduced user uncertainty and would help prevent users from misunderstanding temporary delays that occur while data is being processed as the application not working correctly. This is particularly important for assistive applications, where the user may not visually be able to see the subtle loading indicators and be confused about whether or not the application is currently working.

Another component of usability testing was the quality of the error message communication. If the application did not recognize food item, did not receive food item as input, or the application could not connect to the internet in order to process data, the application was tested based upon its errors and providing descriptive formatting, thus increasing user understanding of the error and providing additional trust in the application. Providing clear descriptive messages will also help with user's frustrations in situations when there are exceptions to the process of using the application. This is paramount to non technical users that do use the application, as using ambiguous and or technical error messages primarily will reduce the usability of the application.

The system was evaluated for perceived independence, one of the project's most important goals. The integration of automation, audible feedback, and user friendly design will enable people with vision loss to get food information independent of visual confirmation or assistance from another person. This independence driven usability outcome is a key focus of the project's assistive objectives and represents a demonstration of real accessibility success beyond just the specific metrics of technical systems.

While usability results were primarily positive, the evaluation also revealed some suggestions for future enhancements. Users might benefit from more voice assisted capture support, haptic feedback signals and or camera

alignment assistance to improve image capture quality in nonvisual capture mode. Despite the fact that this version of the system provides a great deal of accessible use, these enhancements will help increase accessible use levels in difficult real time situations.

The proposed AI Augmented Food Analyzer has been evaluated for usability and accessibility, with positive results for the intended audience of visually impaired users. The application successfully changes AI capabilities into usable assistive devices through simplified workflows, excellent auditory feedback, tactile user interface controls, compatibility with Android accessibility services, clear communication of processing, easy to understand error handling, and independence focused interaction design. There is evidence in the evaluation findings that the developed technology achieves both technology recognition as well as practical accessibility, providing strong evidence of effectiveness for user oriented design for use in real world deployments as assistive technology.

6.6 Performance Under Real World Conditions

Controlled experimental evaluation can provide a lot of useful information regarding your model's accuracy as well as the behaviour of your system under standardised conditions; however, the actual practical value of an assistive intelligent system (AI System) will be determined by the ease and efficiency with which it can be used in uncontrolled environments. Therefore, the AI Augmented Food Analyzer will need to be assessed in a Real World setting, to see how well the integrated application will function in actual use cases that go beyond ideally curated testing data or laboratory style testing scenarios. This part of the evaluation has a large amount of significance because many food recognition systems experience some level

of performance degradation, when deployed out of away from the controlled environment, for reasons such as; inconsistent lighting, variation in the way food is presented, varied levels of background clutter, camera movement instability, as well as noise in the environment. Therefore, the outcome of the real world performance test will be the best indicator of when the developed platform is ready to be deployed.

Testing in real world environments showed that the system continues to possess a high level of recognition performance in normal food presentation situations. When captured food images were taken by users in many of their everyday experiences such as a dining table in their home, on a cafeteria tray, at a restaurant, or just eating in a casual atmosphere at the park; the application tended to provide consistent predictions and helpful nutritional feedback. The system was also able to successfully recognize many different food items despite different plate arrangements, different background textures, and different amounts of ambient light. Therefore, the trained models were confirmed to be generalising beyond the idealised dataset imagery into actual deployment environments.

The findings of the real world evaluation indicate that the system's operational effectiveness is unaffected by moderate levels of environmental variability as measured by varying camera angles, food orientations, plate styles, and other aspects of the dining table. This robustness implies that the augmentation techniques, saliency preprocessing techniques, and various training data that were used during model development functioned to improve the model's ability to generalize beyond strictly standardized visual conditions. It is critically important, therefore, that a practical assistive deployment of this technology is robust so that it can accommodate the non ideal conditions from which users will be capturing images while engaging in their regular daily activities.

Testing performed in real world settings further demonstrated that the integrated YOLOv8 food detection pipeline increased the ability for this technology to be used in a practical manner when there are multiple visible food items in a meal. The testing demonstrated that the detection system successfully localized and identified multiple food areas, in many instances, for various plate combinations and prepared meals, thereby providing greater overall interpretation of a meal than would be provided when using a single label classification. The addition of the aforementioned capabilities to the platform greatly enhances the realism and usefulness of the platform when used as a means of providing daily dietary assistant services.

Even with these attributes, testing under real world conditions has revealed the model will consistently provide less than optimal results in difficult to identify situations when visual features are not properly illuminated. Situations where there is poor lighting compared to bright lights e.g., dark conditions or a strong light source that is very bright e.g. very bright lights vs. there are two sources of high intensity light create occlusion for all of the visual features that are critical to correctly classify an object. When the visual features are occluded, they will not produce clear images, which can lead to less certainty in how to classify the object, as well as an increased probability of misclassification. This is consistent with numerous studies that have been conducted in the computer vision field, which shows that variability in illumination has made working with real world systems for image recognition very challenging.

When food items are also visually challenged, i.e., have either a lesser degree of opacity than adjacent objects or are obscured by utensils, other items etc. this also contributed to a reduction in accuracy. Since the model depends solely on extracting features from the observable areas of food,

when there is a lot of visual blockage there are fewer features available to accurately identify the object, and will increase uncertainty related to classification. Overlapping objects also proved challenging for the entire detection pipeline, as the boundaries of food items will become blurry and or when the overall size of the portion is relatively small compared to the rest of the visual image.

In real world testing, a challenge arose with identifying food categories that looked very similar e.g., two foods that have similar textures, colours or how the food was arranged on the plate. For example, many foods containing similar sauces have different textured grains or have layers, producing less confidence or confusion among the same food categories. This type of ambiguity demonstrates that fine grain food recognition will continue to be an open issue for future food classification research regardless of the sophistication of the food classification techniques currently being used.

The network environment was also found to have an impact on how well the system operated in real world environments, as it uses a distributed client–server architecture i.e., the image is sent to the server to perform the required inferences. When the internet was stable, the application has almost real time responsiveness and works well. Conversely, when the internet was degraded, the time it took to do the recognition of all the images took longer because of how long it took to send the images to the server and the time it took to receive the inference results. The fact that the application functions correctly, even under degraded network conditions, means that the level of responsiveness is directly proportional to the quality of the network. This relationship represents a practical issue that needs to be considered when deploying server based inference architectures.

An accessibility focused real world evaluation of the app showed that it is practical and functional during natural handheld use. The majority of

users were able to take photos successfully and hear vocal feedback without needing to confirm visually, which aligns with the project's assistive objectives. However, the actual quality of the photos continued to be somewhat dependent upon user positioning and framing of their camera in relation to the object being photographed, suggesting that if camera alignment cues or voice backup assistive techniques were added in the future, the usability of the overall app would be further enhanced for real world environments.

The findings confirm that, under realistic conditions of deployment, the system functions as expected though somewhat less well than under controlled testing conditions but nevertheless sufficiently so that assistance can be reasonably expected to occur with most common scenarios where food recognition is the intent of the user. These factors further suggest the training approach, design of the architecture used during deployment, and design of the application work together to create a solid foundation for real world operation.

In summary, real world performance testing of the proposed AI augmented food analyzer demonstrates that the proposed AI augmented food analyzer is successfully moving past controlled experimental benchmarks into deployment within practical real world situations. The food analyzer continues to provide reliable recognition capability in a range of different everyday settings; provides analysis of food meal content; provides users with spoken feedback of analysis results in near real time between the time the food item was scanned and when the user receives the spoken feedback; continues to maintain acceptable operational reliability despite moderate amounts of variation in the environment where it is being used and although the level of performance delivered by the application decreases significantly due to very high levels of visual or connectivity disturbance, the current level of performance delivered by the existing implementation of

the proposed application demonstrates that the application is sufficiently reliable to act as an assistive tool to provide visually impaired users with the ability to identify food items in their daily lives. These results provide validation of the deployment focused design intent of the project and establish the practical feasibility of the proposed intelligent food analysis solution.

6.7 Limitations and Error Analysis

The AI Augmented Food Analyzer project is successful on many levels, but there are still limits to what it can do due to design and construction constraints. It's important to identify limitations to provide an accurate review of the platform, as well as to inform ongoing research and improvements to the system. The proposed system shows strong performance as expected given its trained design but still exhibits certain technical, architectural, and practical limitations due to the limitations in its datasets, model generalization limits, architectural deployment limitations, and variability associated with operating in a real world environment.

One of the main limitations of the current system is that it requires a predefined set of trained food types. Only food classes that were present in the training dataset and the corresponding nutritional database are capable of having their recognition models developed. Therefore, if the user presents food that is not in the trained set, regionally specific food items that are not included in the training set, or newly developed food items, the system may incorrectly classify the food by identifying it as a similar type of known food or will fail to provide good recognition. Most supervised deep learning classification systems have this limitation with respect to closed set recognition. The closed set means that the knowledge

within the model is bounded by its training set.

The quality of the food dataset is another major limitation. Although large benchmark datasets such as Food101 and USDA based nutritional resources provide substantial training material, many existing food datasets emphasize isolated or single dish imagery rather than highly diverse real world meal scenes. As a result, model generalization to complex mixed meal environments remains constrained by the representativeness of available training data. The absence of sufficiently large and diverse annotated datasets for multi food, real world meal recognition limits the achievable robustness of current recognition models.

The system also remains sensitive to challenging visual conditions, including extreme lighting imbalance, severe shadows, glare, heavy occlusion, poor camera focus, motion blur, and visually cluttered backgrounds. Although augmentation strategies and saliency based preprocessing improve robustness, they cannot eliminate the fundamental dependence of visual recognition models on sufficient image quality and distinguishable visual features. Under highly degraded visual input conditions, recognition reliability decreases predictably.

Another limitation concerns the difficulty of fine grained food differentiation. Many food types share visual similarities despite having different semantics and therefore will be difficult for even advanced deep learning models to distinguish between classes of different semantics if there is a strong visual overlap without a contextual or multimodal input.

This limitation is presented as a greater challenge for fine grained visual recognition research than just an issue with the proposed system.

From a nutritional analysis perspective, the current system produces approximate calorie and ingredient values using standardized mappings of food categories; therefore these values represent the average calorie and in-

gredient contributions for that food category rather than specific nutrition for the portion size of each food ingesting. Currently, there is no estimate of the portion size detected in the image; therefore, the output values represent average or generalized values rather than actual measured values. As these values still have value as helpful diet information, user requiring highly accurate nutritional monitoring may find the output values do not have sufficient accuracy.

The restrictions imposed by network dependency in the deployment of server side architecture are another significant downside. Inference has to occur on the backend server and, as such, when a stable connection is not available, the system experiences negative consequences and limitations in its usability during those times. Therefore, even though the ability to perform inference is operationally independent of a stable Internet, if there is no successful backend connectivity, then at least one of the user experience components will be negatively affected.

Latency, due to variable network conditions also contributes to variability in the practical responsiveness of this type of deployment. While generally acting as an optimized deployment mechanism, the overall system timeline of providing near real time feedback with respect to inference will be subject to variable upload bandwidth, variable server availability, and variable communication latencies that are inherently present in a client server architecture due to the distributive nature of the architecture. These factors also have a negative effect on the user experiences of the users using the system while they are experiencing poor connectivity.

A third limitation involves the dependent aspects of the camera framing and user capture. It can be very difficult for individuals with visual impairments to calibrate the camera to be appropriately located for the recognition to occur due to the absence of visual feedback, which may result

in decreased performance in the recognition of the food product because of sub optimal framing of the food product within the camera, capturing the food product in its entirety within the camera viewfinder, and or capturing the food product with respect to proper camera orientation. While the existing design for accessibility has significantly reduced operationally burdensome design and operation of the system, the use of additional assistive capture guidance mechanisms to aid in establishing better input quality and provide consistent results should be evaluated for inclusion at this stage of the development cycle.

At present, the deployed models are static after training; they do not adaptively learn from user interaction or from the provision of new types of food. There will be no ongoing adaptation of this system to reflect changes in definition until such time as the models are either retrained or upgraded at the backend of the system, while changes to the underlying definition will require a new development cycle and approval process.

In addition, it is important to recognize that limitations characteristic of all AI systems will also constrain theAI Augmented Food Analyzer. The recognition component of this platform operates on the basis of complex statistical algorithms and, as such, decision making remains non transparent, even though they produce high levels of empirical accuracy. In addition, AI systems may produce unexpected results when attempting to categorize objects that fall outside their training sets.

As such, although the proposed AI Augmented Food Analyzer has limitations, it remains a substantial advance over existing assistive technology for food recognition and should not be discounted because of the limitations of its design or function. Rather, these limitations will help define the current capabilities of the platform and will show where there is opportunity for additional impact through research and development. Acknowledging

constraints candidly will support an accurate understanding of current capabilities while establishing a clear path forward for further development and expansion.

Chapter 7

Conclusion

To develop an AI augmented food analyzer, our goal was to create a new way to assist visually impaired individuals by enabling them to identify food as well as understand its nutritional value. By using advanced artificial intelligence technologies, such as deep learning, computer vision, and app development, the project represents a fusion of cutting edge technology with real world applications. The entire pipeline for creating this solution—from image acquisition and preprocessing through food classification and counting to delivering output in a usable format—was developed and implemented. To identify food using multiple deep learning architectures, including ResNet101, EfficientNet, ConvNeXt, and MobileNetV3Large models, we evaluated these architectures to determine which worked best for food identification. In addition, YOLOv8 was incorporated into the solution to provide the ability to identify multiple food items from one image, thereby overcoming one of the key limitations of traditional image classification based solutions. Robustness was improved by employing saliency techniques along with data augmentation strategies on the system. These methods guide the model's attention to the pertinent parts of images and expose the model to a variety of visual environments for reliable performance in real world scenarios. Further, negative samples from the COCO dataset aided in building the model's ability to differentiate food and nonfood items decreasing false positives. The client server architecture provides the ability for the system to utilize deep learning models on a private server while maintaining a lightweight mobile application enabling efficient processing and scalability allowing for practical deployment of the system. The evaluation results reflect a good balance between accuracy, efficiency and usability by the system. A major contribution of the project is its accessible focused design. The proposed system has been designed specifically for visually impaired users, unlike traditional

food recognition applications. The system communicates the results of the food recognition through auditory feedback using text to speech capability integrated into the system. A mobile application was developed using Android Studio with a simple and easy to use interface with minimal interaction by the user while providing real time performance. Although this system has made great progress, it still has some limitations: the training data set restricted the number of food categories to those available in the training data set and therefore limits what types of new food, the system trained to recognize; the estimates of calories are based on rounded values and do not consider the size or preparation method of the portion; and the types of environments in which the food will be recognized, such as poor lighting or occlusion of the object, may have an effect on the recognition accuracy of the food. Addressing these limitations is a viable area for future work. Increasing the number of food categories in the data set, especially with respect to regional foods and foods from different cultures, is one area that can be improved upon. By improving the generalization ability of the model through using more complex training methods and larger training data sets will allow for a significant improvement in the recognition accuracy of the food that the system recognizes. An area for improvement for this system is to add depth estimation or portion size analysis so that the calorie estimates would be based on a food item's actual size in conjunction with its ingredients and preparation. By estimating calories based on actual measurements instead of rounded values, the end user would be able to make more informed decisions about their nutritional intake. Improving the performance of this system in offline mode could reduce the reliance on Internet connectivity so that this application would operate without interruptions or depend on continuous connectivity. The development of future applications must consider the limitations of datasets,

particularly for multi food scenarios. For example, the USDA dataset is currently designed to identify nutrients for individual foods found in an image. However, if datasets were to be expanded to include multiple foods "objects" together that are included in the same image, it would help make the system more accurate and contextually aware of the nutrients present in both individual and multiple food types. This type of implementation could ultimately lead to developments in the system's ability to integrate multi object detection i.e., identifying multiple objects in a scene with full nutritional mapping i.e. having as much information about each food as possible. As such there would be a significant increase in the ability of the system to accurately process complex meal combinations. Another potential area of improvement is creating more functionality with accessibility features using a voice command interface and providing multi language support, allowing for more natural interaction with the system. Continuous improvement of user interface and user feedback mechanisms can result in improved usability and confidence in using the system. In summary, the AI Aided Food Analyzer illustrates how artificial intelligencecan be leveraged to develop important assistive technologies. By applying deep learning toward a user centered design methodology, it creates not only a technically successful product, but also an increase in independence and quality of life for visually impaired persons. The future of this research will lead to further advancements in a unified solution to intelligent assistive technology development.

References

[1] World Health Organization. *World Report on Vision*. World Health Organization, Geneva, Switzerland, 2019.

[2] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[3] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[4] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[5] J. Harel, C. Koch, and P. Perona. Graph-based visual saliency. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 545–552, 2007.

[6] Z. Liu, H. Mao, C. Wu, C. Feichtenhofer, T. Darrell, and S. Xie. A convnet for the 2020s. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11976–11986, 2022.

[7] A. Howard et al. Searching for mobilenetv3. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1314–1324, 2019.

[8] L. Bossard, M. Guillaumin, and L. Van Gool. Food-101 – mining discriminative components with random forests. In *Proceedings of the*

European Conference on Computer Vision (ECCV), pages 446–461, 2014.

[9] T. Lin et al. Microsoft coco: Common objects in context. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 740–755, 2014.

[10] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pages 234–241, 2015.

[11] S. Woo, J. Park, J. Lee, and I. Kweon. Cbam: Convolutional block attention module. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 3–19, 2018.

[12] M. Cheng, G. Zhang, N. Mitra, X. Huang, and S. Hu. Global contrast based salient region detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(3):569–582, 2015.

[13] L. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 801–818, 2018.

[14] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016.

[15] G. Jocher et al. Yolov8, 2023. [Online]. Available: https://github.com/ultralytics/ultralytics.

[16] A. Shorten and T. M. Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):1–48, 2019.

[17] E. Cubuk, B. Zoph, J. Shlens, and Q. Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 702–703, 2020.

[18] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, New York, NY, USA, 2006.

[19] E. Breck, S. Cai, E. Nielsen, et al. The ml test score: A rubric for ml production readiness. In *IEEE International Conference on Big Data*, 2017.

[20] G. Van Rossum and F. Drake. *Python Reference Manual*. PythonLabs, 2001.

[21] A. Paszke, S. Gross, F. Massa, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 8024–8035, 2019.

[22] J. Nielsen. *Usability Engineering*. Morgan Kaufmann, 1994.

[23] M. Hassenzahl. *Experience Design: Technology for All the Right Reasons*. Morgan and Claypool, 2010.

[24] D. Norman. *The Design of Everyday Things*. Basic Books, 2013.

[25] K. Wiegers and J. Beatty. *Software Requirements*. Microsoft Press, 2013.

[26] J. Lazar. *Accessibility and Usability of Information Systems*. Elsevier, 2015.

Appendix A

User Manual

The user can open the application by using Google Assistant voice commands. After opening the application, the user is provided with options including Scan Food, User Guide, History, and Close App. The Scan Food option captures the food image through the camera and analyzes it using AI models to identify the food item and estimate its ingredients and calorie information. The application displays the results on the screen and also speaks them aloud for user convenience. The User Guide option explains how to operate the application, while the History option stores previously scanned food records for future reference.

Asim Altaf Shah

Final_Thesis_12_05_2026-1-80

Paper check

Document Details

Submission ID
trn:oid:::3618:138759807

Submission Date
May 13, 2026, 9:58 AM GMT+5

Download Date
May 13, 2026, 10:21 AM GMT+5

File Name
Final_Thesis_12_05_2026-1-80.pdf

File Size
1.6 MB

80 Pages

20,859 Words

118,447 Characters

3% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography

Match Groups

63 Not Cited or Quoted 3%
Matches with neither in-text citation nor quotation marks

0 Missing Quotations 0%
Matches that are still very similar to source material

0 Missing Citation 0%
Matches that have quotation marks, but no in-text citation

0 Cited and Quoted 0%
Matches with in-text citation present, but no quotation marks

Top Sources

1% Internet sources
0% Publications
3% Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

Match Groups

- **63** Not Cited or Quoted 3%
  Matches with neither in-text citation nor quotation marks
- **0** Missing Quotations 0%
  Matches that are still very similar to source material
- **0** Missing Citation 0%
  Matches that have quotation marks, but no in-text citation
- **0** Cited and Quoted 0%
  Matches with in-text citation present, but no quotation marks

Top Sources

1% Internet sources
0% Publications
3% Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1	Student papers	Clarkston Community Schools on 2023-05-05 <1%
2	Student papers	University of Nottingham on 2024-05-31 <1%
3	Student papers	The University of Manchester on 2025-04-11 <1%
4	Student papers	Higher Education Commission Pakistan on 2013-12-13 <1%
5	Student papers	King Fahd University for Petroleum and Minerals on 2024-12-17 <1%
6	Student papers	University of Huddersfield on 2023-01-30 <1%
7	Publication	"Accelerating Discoveries in Data Science and Artificial Intelligence I", Springer Sc... <1%
8	Student papers	University of Ulster on 2018-04-10 <1%
9	Internet	eprints.utem.edu.my <1%
10	Internet	ir.uitm.edu.my <1%

11 Student papers
University of Westminster on 2025-07-11 <1%

12 Student papers
Heriot-Watt University on 2025-04-17 <1%

13 Publication
Lev V. Eppelbaum, Olga Khabarova, Michal Birkenfeld. "Advancing archaeo-geop... <1%

14 Student papers
Coventry University on 2026-04-20 <1%

15 Student papers
University of Strathclyde on 2023-08-13 <1%

16 Student papers
Liverpool John Moores University on 2023-03-27 <1%

17 Student papers
University of Wollongong on 2025-03-26 <1%

18 Student papers
University of Sheffield on 2024-09-01 <1%

19 Internet
diva-portal.org <1%

20 Student papers
Heriot-Watt University on 2025-03-27 <1%

21 Student papers
University of East London on 2012-12-02 <1%

22 Internet
pdffox.com <1%

23 Internet
perfectcustompapers.com <1%

24 Student papers
Higher Education Commission Pakistan on 2013-12-13 <1%

25 Student papers
Higher Education Commission Pakistan on 2014-05-05 <1%

26 Student papers
University of Technology, Sydney on 2025-10-06 <1%

27 Student papers
VIT University on 2025-04-03 <1%

28 Internet
goo.by <1%

29 Internet
pastovu.vdu.lt <1%

30 Internet
scientific-research.kiwiinnovate.com <1%

31 Internet
www.researchgate.net <1%

32 Publication
Anthimopoulos, Marios M., Lauro Gianola, Luca Scarnato, Peter Diem, and Stavro... <1%

33 Student papers
Higher Education Commission Pakistan on 2015-12-23 <1%

34 Student papers
Higher Education Commission Pakistan on 2022-11-01 <1%

35 Student papers
Middlesex University on 2023-04-30 <1%

36 Student papers
UOW Malaysia KDU University College Sdn. Bhd on 2025-12-01 <1%

37 Student papers
University of Hertfordshire on 2025-04-04 <1%

38 Publication
Xinda Liu, Yaohui Zhu, Linhu Liu, Jiang Tian, Lili Wang. "Feature-SuppressedContr... <1%

39 Publication

Yi Lu, Dongyan Wei, Hong Yuan. "Magnetic Localization Method for Vehicles Base... <1%

40 Internet

www.aminer.org <1%

41 Internet

www.techscience.com <1%

42 Student papers

Higher Education Commission Pakistan on 2026-04-28 <1%

43 Student papers

Walden University on 2012-02-06 <1%

44 Student papers

University of Nottingham on 2025-05-03 <1%

Asim Altaf Shah

Final_Thesis_12_05_2026-1-80

Paper check

Document Details

Submission ID
trn:oid:::3618:138759807

Submission Date
May 13, 2026, 9:58 AM GMT+5

Download Date
May 13, 2026, 10:21 AM GMT+5

File Name
Final_Thesis_12_05_2026-1-80.pdf

File Size
1.6 MB

80 Pages

20,859 Words

118,447 Characters

*% detected as AI

AI detection includes the possibility of false positives. Although some text in this submission is likely AI generated, scores below the 20% threshold are not surfaced because they have a higher likelihood of false positives.

Caution: Review required.

It is essential to understand the limitations of AI detection before making decisions about a student's work. We encourage you to learn more about Turnitin's AI detection capabilities before using the tool.

Asim Altaf Shah

Final_Thesis_12_05_2026-81-160

Paper check

Document Details

Submission ID
trn:oid:::3618:138763790

Submission Date
May 13, 2026, 10:17 AM GMT+5

Download Date
May 13, 2026, 10:23 AM GMT+5

File Name
Final_Thesis_12_05_2026-81-160.pdf

File Size
16.2 MB

80 Pages

21,123 Words

121,594 Characters

1% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography

Match Groups

22 Not Cited or Quoted 1%
Matches with neither in-text citation nor quotation marks

0 Missing Quotations 0%
Matches that are still very similar to source material

0 Missing Citation 0%
Matches that have quotation marks, but no in-text citation

0 Cited and Quoted 0%
Matches with in-text citation present, but no quotation marks

Top Sources

0% Internet sources
0% Publications
1% Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

Match Groups

- **22** Not Cited or Quoted 1%
  Matches with neither in-text citation nor quotation marks
- **0** Missing Quotations 0%
  Matches that are still very similar to source material
- **0** Missing Citation 0%
  Matches that have quotation marks, but no in-text citation
- **0** Cited and Quoted 0%
  Matches with in-text citation present, but no quotation marks

Top Sources

0% Internet sources
0% Publications
1% Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1	Student papers	Universiti Tun Hussein Onn Malaysia on 2026-02-05 <1%
2	Student papers	BB9.1 PROD on 2026-04-22 <1%
3	Internet	dirros.openscience.si <1%
4	Student papers	University of Birmingham on 2022-09-15 <1%
5	Student papers	University of Southampton on 2016-09-14 <1%
6	Publication	Aymen A. Altae, Abdolvahab Ehsani Rad, Keyvan Mohebbi. "Advanced COVID-19 d... <1%
7	Publication	M. K. Kavitha Devi, Thangavel Murugan, K. Indira, Raja Lavanya, S. Abinaya. "Intel... <1%
8	Publication	Nigon, Tyler John. "Uncertainty in Economic Optimum Nitrogen Rate and Accurac... <1%
9	Student papers	University of Westminster on 2025-04-16 <1%
10	Internet	krishikosh.egranth.ac.in <1%

11 Student papers

Liverpool John Moores University on 2022-03-03 <1%

12 Publication

Harvinder Singh, Priyanka Kaushal, Sarabjeet Kaur. "Advanced Computing and AI... <1%

Asim Altaf Shah

Final_Thesis_12_05_2026-81-160

Paper check

Document Details

Submission ID
trn:oid:::3618:138763790

Submission Date
May 13, 2026, 10:17 AM GMT+5

Download Date
May 13, 2026, 10:24 AM GMT+5

File Name
Final_Thesis_12_05_2026-81-160.pdf

File Size
16.2 MB

80 Pages

21,123 Words

121,594 Characters

*% detected as AI

AI detection includes the possibility of false positives. Although some text in this submission is likely AI generated, scores below the 20% threshold are not surfaced because they have a higher likelihood of false positives.

Caution: Review required.

It is essential to understand the limitations of AI detection before making decisions about a student's work. We encourage you to learn more about Turnitin's AI detection capabilities before using the tool.

Junaid Imtiaz

Final_Thesis_12_05_2026-161-232

Paper check

Document Details

Submission ID
trn:oid:::3618:138761842

Submission Date
May 13, 2026, 10:02 AM GMT+5

Download Date
May 13, 2026, 10:22 AM GMT+5

File Name
Final_Thesis_12_05_2026-161-232.pdf

File Size
5.6 MB

72 Pages

19,011 Words

108,267 Characters

2% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography

Match Groups

36 Not Cited or Quoted 2%
Matches with neither in-text citation nor quotation marks

0 Missing Quotations 0%
Matches that are still very similar to source material

0 Missing Citation 0%
Matches that have quotation marks, but no in-text citation

0 Cited and Quoted 0%
Matches with in-text citation present, but no quotation marks

Top Sources

1% Internet sources
1% Publications
1% Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

Match Groups

- **36** Not Cited or Quoted 2%
  Matches with neither in-text citation nor quotation marks
- **0** Missing Quotations 0%
  Matches that are still very similar to source material
- **0** Missing Citation 0%
  Matches that have quotation marks, but no in-text citation
- **0** Cited and Quoted 0%
  Matches with in-text citation present, but no quotation marks

Top Sources

1% Internet sources
1% Publications
1% Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1	Publication	Ervin Sejdić, Tiago H. Falk. "Signal Processing and Machine Learning for Biomedic... <1%
2	Internet	www.thefreelibrary.com <1%
3	Student papers	University of Liverpool on 2024-04-18 <1%
4	Student papers	University of Bradford on 2026-04-07 <1%
5	Student papers	Higher Education Commission Pakistan on 2017-12-05 <1%
6	Publication	Dave, Namrata Ashokbhai. "Content Based Video Retrieval from Gujarati News Vi... <1%
7	Student papers	The University of Manchester on 2013-09-02 <1%
8	Internet	d-nb.info <1%
9	Internet	krex.k-state.edu <1%
10	Internet	www.ijstr.org <1%

11 Internet
backend.orbit.dtu.dk <1%

12 Internet
blog.maxant.co.uk <1%

13 Internet
repositum.tuwien.at <1%

14 Publication
Hassen Fourati, Krzysztof Iniewski. "Multisensor Data Fusion - From Algorithms a... <1%

15 Student papers
Imperial College of Science, Technology and Medicine on 2020-10-08 <1%

16 Publication
Xiang, Lim Wei. "Diabetic Retinopathy Image Classification with Quantitative Sali... <1%

17 Student papers
unistgallen-plagiat on 2024-02-27 <1%

18 Internet
www.eurchembull.com <1%

19 Student papers
ABV-Indian Institute of Information Technology and Management Gwalior on 202... <1%

20 Student papers
National Institute of Technology, Warangal, Telangana, India on 2026-04-27 <1%

21 Student papers
National University of Ireland, Galway on 2021-03-16 <1%

22 Publication
Shalli Rani, Ayush Dogra, Ashu Taneja. "Smart Computing and Communication fo... <1%

23 Student papers
TH Köln on 2025-11-28 <1%

24 Student papers
Teesside University (Blackboard Ultra) on 2026-04-30 <1%

25 Publication

Yanai, Keiji, and Yoshiyuki Kawano. "Food image recognition using deep convolut... <1%

26 Internet

iapress.org <1%

27 Internet

impa.usc.edu <1%

28 Internet

jtrpinto.github.io <1%

29 Internet

leonarmo.folk.ntnu.no <1%

30 Internet

www.cvast.tuwien.ac.at <1%

31 Internet

www.mdpi.com <1%

32 Student papers

The Indian Institute Of Management And Engineering Society on 2026-05-04 <1%

33 Student papers

Liverpool John Moores University on 2025-02-23 <1%

Junaid Imtiaz

Final_Thesis_12_05_2026-161-232

Paper check

Document Details

Submission ID
trn:oid:::3618:138761842

Submission Date
May 13, 2026, 10:02 AM GMT+5

Download Date
May 13, 2026, 10:23 AM GMT+5

File Name
Final_Thesis_12_05_2026-161-232.pdf

File Size
5.6 MB

72 Pages

19,011 Words

108,267 Characters

*% detected as AI

AI detection includes the possibility of false positives. Although some text in this submission is likely AI generated, scores below the 20% threshold are not surfaced because they have a higher likelihood of false positives.

Caution: Review required.

It is essential to understand the limitations of AI detection before making decisions about a student's work. We encourage you to learn more about Turnitin's AI detection capabilities before using the tool.