AN INTELLIGENT FOOD INVENTORY MONITORING SYSTEM USING MACHINE LEARNING AND COMPUTER VISION

Tianyu Li[1] and Yu Sun[2]

[1]St. George's School, 4175 W 29th Ave, Vancouver, BC V6S 1V1, Canada
[2]California State Polytechnic University,
Pomona, CA, 91768, Irvine, CA 92620

ABSTRACT

Due to technological advancements, humans are able to produce more food than ever before. In fact, the food production level is so high that all population could be supported if the food resource is distributed correctly. Yet, it is more than common to see items left expiring on the supermarket shelves, wasting the food resource that could otherwise be useful. Neither are the adverse impacts on the climate due to food disposal in anyone's favor or interest. This paper proposes an application to identify the stock status of supermarket items, specifically food items, so that supermarket managers can react to the selling status and prevent oversupply. The key tool implemented in the application is computer vision, specifically YOLOv5, which uses convolutional neural networks [1]. The model automatically recognizes and counts the items in a taken picture. We applied our computer vision model to numerous supermarket shelf photos and conducted an evaluation of the model's precision and speed. The results show that the application is a useful tool for users to log supermarket stock information since the computer vision model, despite lacking slightly in object detection precision, can return a reliable count for well-taken photos. As a platform where such information is shared, the application is therefore a viable tool for store managers to import amounts of food accordingly and for the public to be informed and make smart buying choices.

KEYWORDS

Flutter, YOLOv5, Computer Vision, Inventory Management.

1. INTRODUCTION

Food waste is detrimental, especially in this environmentally stressed world: not only the food itself but the expended resources and energy – including irrigation and transportation – are wasted. One of the main culprits is the supermarket; among the 931 million tons of food waste each year, 13% comes from retail [2]. Exacerbating the problem, 44.7% of food waste in 2020 will be diverted to landfill, the least favorable disposal method [3]. Food landfills produce a large amount of methane, an extremely potent greenhouse gas; about 7% of global greenhouse gas is due to preventable food waste [4]. Besides, the estimated annual food waste cost in the North American economy is about $278 billion [5]. As seen above, food waste is both an economically tolling and environmentally-damaging problem that affects virtually everyone in the world.

In order to eliminate the problem, it is therefore most favorable to limit the amount of food waste. There have been countless instances of fully loaded supermarket shelves of items not being sold anywhere before their expiration, by which time the retailers have no choice but to discard the

David C. Wyld et al. (Eds): ARIA, SIPR, SOFEA, CSEN, DSML, NLP, EDTECH, NCWC - 2022
pp. 145-155, 2022. CS & IT - CSCP 2022 DOI: 10.5121/csit.2022.121512

food products. This could be attributed to either managers losing track of consumers' demand or consumers not knowing where the items are available at. The proposed solution solves the problem by offering both sides foresight into a supermarket's supply. The application increases the transparency of food stocks in the supermarket and automates the process using computer vision. Managers can then monitor their inventories according to the demands and adjust to the data over time.

Many techniques have been implemented to limit the amount of food waste in grocery stores. One commonality among many of them is the usage of technology. Some supermarkets use technology as a means to track the expiration dates of products. This allows the retailers to discount the near-expired products and earn profits from that, and technology saves retailers time as it automates the process [6]. Others use technology to digitalize the layout of their stores. Information of products in the inventory are then more accessible, and retailers will no longer have to rely on intermediaries when importing from warehouses, decreasing the amount of perishables that previously exist due to inefficiencies in handling [7]. Yet, such ideas are still generally tentative as they are currently being experimented with in a few stores.

Besides technology, grocery stores can adopt alternative supply practices. To start with, they can partner with their suppliers by actively communicating consumers' demand for the different food items. Some agri-tech companies have further supported this collaboration by sharing different retailers' market information with farmers on application software [8]. This ultimately allows farmers to plan their production better and make use of potential wastes, such as producing energy [9]. However, the communication between stores and farms can sometimes be inefficient as there could be a time-lag when stores receive products, and some stores' tracking of items is inaccurate.

Another popular waste-reduction method is making use of all products. Instead of rejecting the imperfect-looking food, which are food that are not as good as others in appearance, supermarkets promote them. For instance, grocery stores like Morrison's sell these foods at discounted prices. In addition, instead of discarding surplus produce into landfill, supermarkets convert them into produce or distribute them to people who need the food [10]. These methods vitally take advantage of the food produced, but the key shortcoming is that they are short-term practices that require a high level of civilians' stewardship, which may be hard to achieve in some communities.

In this paper, our proposed solution is a real-time digital grocery stock tracking system that provides counts and images of supermarket items, which can be provided by the users of the application. The application shows a list of supermarkets and collects stock information of food items in the supermarkets. Users can see the exact location of a desired item by clicking a supermarket. In addition, users can plan their purchase by searching for certain items. The item stock information at different store locations will be retrieved, and users can select the number of items and where to purchase in their trip.

This proposed solution encompasses all the existing methods. The application is a technology that practically digitalize grocery stores. Consumers can have a more holistic perspective of items in different stores around them. They can also help the stores update inventories by simply taking a photo, which is then analyzed by AI. Store managers can have clear data on their items. This makes their communication with the suppliers much more efficient since the trends of the data allow their future imports of goods to be more oriented. Therefore, by providing food items at amounts that suit the consumers' needs, supermarkets can reduce the amount of perishables that would be left expired and wasted but make use of them fresh instead.

In two application scenarios, we demonstrate how implementing the computer vision model as an integral functionality into the application increases both the utility and usability of the application through its efficiency in updating stock information. First, the accuracy of the computer vision model is evaluated in two components – object detection and counting. The model labeled a set of validation and test images, and its performance in object detection was measured by four key metrics – precision, Mean Average Precision (mAP), recall, and General Intersection Over Union (GIOU). While the returned labels were slightly less accurate, the model could localize the items to operate on. Moreover, the model's counts in several cases grocery images were manually checked over. It accurately detected and counted items in large-scale, upright-angled, and high-resolution pictures. By taking proper item-specific photos, users can update the inventory accurately and more quickly for their local communities with the numbers returned by the embedded computer vision model.

Second, the speed of the computer vision model is gauged through the timeit module in Python. The model, which runs on a backend server, is tasked with 60 photos of various supermarket items, and the execution time of the model in counting items in each photo is recorded. Key statistical parameters, such as mean and standard deviation, are calculated and convincing that the computer vision model is quick in counting items. By providing a count in a timely fashion, the computer vision model makes updating the inventory convenient for users, and more people will, therefore, be inclined to download the application and limit food waste.

The rest of the paper is organized as follows: Section 2 outlines and details the challenges in developing the computer vision and application; Section 3 elaborates on how the proposed solution works; Section 4 presents experiments that test the efficiency of the solution and analyzes the solution; Section 5 lists related works that address the problem of grocery food waste; Section 6, finally, gives the conclusion remarks and points out future works of this project.

2. CHALLENGES

To develop a helpful, user-friendly application system, several challenges have been identified as follows.

2.1. Designing the layout and functions of the application

The key to the success of an application is its user-friendliness. This characteristic is often shown through how easily the users can navigate through the application and the functions that users need to benefit from the application. As a result, a lot of efforts have been dedicated to designing how all the stock information should be presented. This process is the most time-consuming as multiple possibilities of the layouts were tested before finalizing on one. Eventually, the dashboard shows all stores with elapsed information in connected pages. The camera option is embedded inside the item page so that the users can take a photo of the stock when looking for the items; this also makes the most sense because the information analyzed from the photo can then most easily be directed to the according location in the application. To further improve, perhaps including a business analysis for managers or a wishlist for consumers can build a community for the application.

2.2. Accommodating all the various grocery items in the dataset

In an average supermarket, there are about 40000 products on the innumerous shelves [11]. What differentiates the products are what the products are and the producing or manufacturing company, which sometimes is one of the consumers' decision factors. Theoretically, in order for

the machine to recognize and discern between the products, it has to be trained with all the products, which will require a large amount of data. Feeding in such a large dataset is already an almost impossible task, and the increased variety of products simply makes the model more prone to errors and inaccuracies when identifying and counting the items. As a result, it is impractical to attempt to apply a model to every item. Alternatively in the solution, most similar items are regrouped to one general group. For instance, all bread items – including whole wheat, white, or grain – are all under one category of bread, which is trained with images of the various sub-products. This ensures that while there is less clue as to the specifics of the items, the products can still be identified and counted.

2.3. Creating a secure server for the application

The computer vision model embedded in the camera function is a central feature that processes inventory photos, and it is hosted on an online server in the backend. Since the model should operate whenever a photo is taken, the server needs to be accessible at all times. An HTTP server was first deployed on Repl.it for the application because it leverages the Flask web framework, which comes with established libraries and is flexible for implementation. When testing with real-time photos, however, the server, despite running autosynchronously at first, had preserving connection and compatability issues, which crashed the application and rendered it counterproductive. Consequently, the model was integrated into the Amazon Web Services (AWS) cloud server instead. The AWS server offered a stable connection once launched, and it efficiently returns object detection and counting outputs to the applicaiton user interface (UI) upon requests. This keeps the application functional as users can update inventories by handily taking photos using the camera.

3. SOLUTION

This application provides users with the amount, location, and image of different food items in grocery stores of close geographical proximity. As shown in Figure 1, the system is composed of three main pages to achieve the purpose: Stores, Wishlist, and Me.

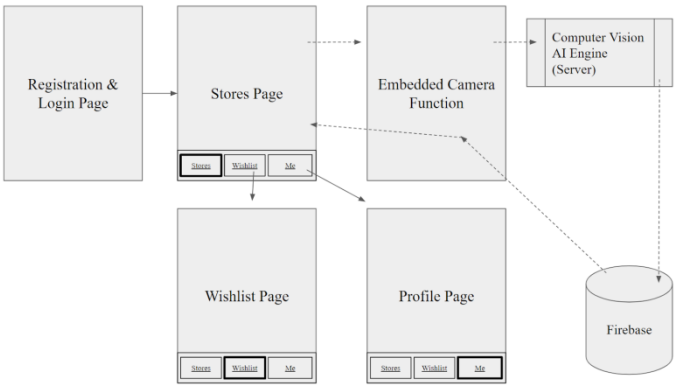

Figure 1. Application Overview

In the “Stores” page, users can access all the aforementioned information by clicking through each store or typing the desired items in the search bar above. Items of names related to the one typed in the search bar will be identified and included in the drop down menu for selection. The camera function embedded in the stores page is the only means to update the inventory. After taking a photo of one type of item at a time, the user specifies the item, and the AI counts and

returns the number of items in the photo to both the database and the application UI. If incorrect, users can edit the number. The finalized number is then updated.

The “Wishlist” page serves for planning purposes. For supermarket managers, the list could be filled with items that they wish to import from their suppliers; for consumers, the list could be filled with items that they wish to buy for daily usage. An image of the item’s stock is presented when populating the list.

The “Me” page displays the user’s registration information and allows editing of it. The credential to log into the system consists of an email address and a password, which could be set after clicking the “Register” option.

The entire application is developed in Flutter using the programming language Dart and is available in both Android and iOS devices. The key data that allows the functionality of the application – number and photo of food items and user profile – are all stored in the cloud database Google Firebase. The computer vision technology of the camera function hosts on a server and has been custom trained on a manually created dataset in Roboflow with YOLOv5, a helpful object detection algorithm that uses the convolutional neural network. The algorithm divides a given image into grids, or kernels, performs pooling to extract dominant features, and calculates an expected probability for each component. It repeats the above procedure and returns the final object detection output after non-maximum suppression, which prevents false-positive identifications.

In short, the “Stores” page contains the core feature of accessing and updating the stock information of food items in grocery stores to the backend database and the application; the camera function supports the update feature with computer vision that counts items in a taken photo. The “Wishlist” page documents users’ personal demands, users can manage their accounts on the “Me” Page.

The flow of the “Stores” page follows the stores’ physical layout order, as shown in Figure 2. A list of stores are first presented. In each store, there are different aisles, and each aisle houses several shelves, on which users can find the specific items with their quantities. A photo of an item can be found by clicking on the item on the list. Users direct through these components by clicking on the desired entry on each subpage. Alternatively, users can directly find an item by typing the item name in the search bar and clicking on it from the dropdown menu. The items’ stock information in each store is retrieved for users to view.

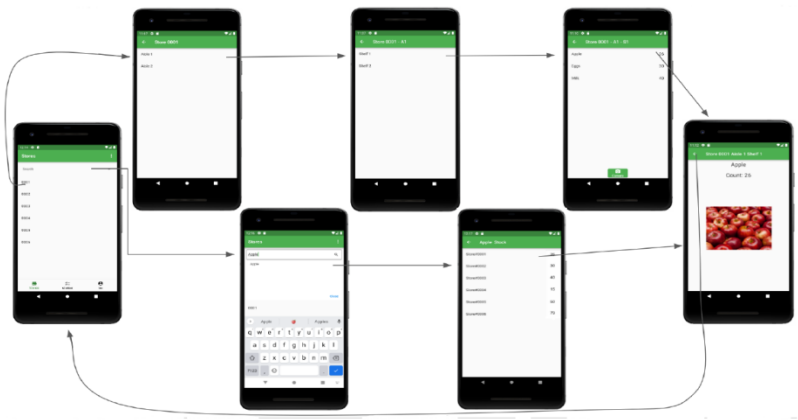

Figure 2. Stores Page UI

Clicking on the “Camera” button at the bottom of the “Items” subpage, users can activate the camera function. A count will be automatically returned once the user successfully takes a photo of items. To complete the update, users just have to confirm the food item and number of items the AI identified. After that, the data will be retrieved to Firebase and shown on the application UI.

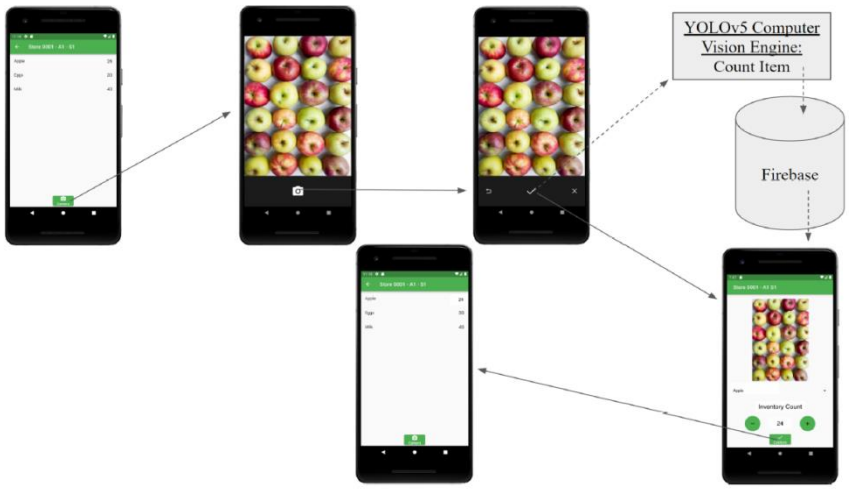

Figure 3. Camera Function UI

Lastly, the “Wishlist” page mostly leverages the search bar functionality, as shown in Figure 4. Users can find the quantity of their desired item at different stores by typing them in the search bar. After selecting the item in a desired store, the item’s most recent photo will be displayed along with the quantity on a new page, where users can select the number of the item they want. The item can then be added to the wishlist and are still editable.

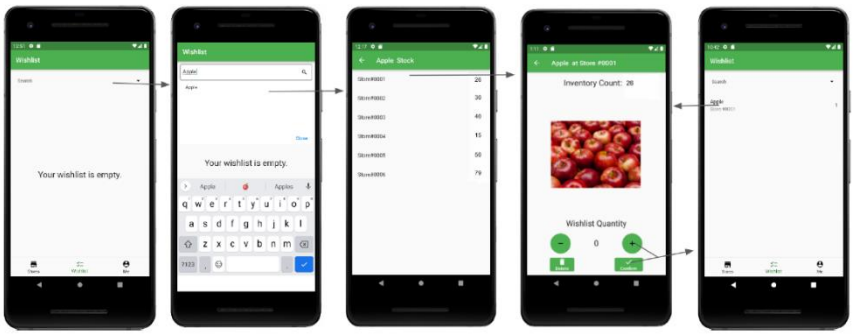

Figure 4. Wishlist Page UI

When a user edits a wishlist item, the stock of the item could be updated, and how many of the item the user wants is also constantly modified. As a result, these two pieces of information are asynchronous, requiring real-time updates, and thus, a StreamBuilder, shown in Figure 5, is implemented [12]. The StreamBuilder widget treats these real-time data as streams, and since these data are stored in Firebase, it takes a stream that constantly requests a snapshot of the user’s wishlist in the Firebase. The builder, which creates the UI, default returns “Loading” text when checking stream snapshot data. If the snapshot data is erroneous, an error message is returned. Otherwise, the builder does further checks. If the user already had a wishlist quantity for the item

in the designated store, the builder builds the page with the item's stock, which is retrieved in another StreamBuilder, and the user's wishlist quantity, which will be updated by constantly undergoing the same checks as the user edits it. If not, the builder defaults return the item's stock and 0 as the user's wishlist quantity for the user to edit.

```
StreamBuilder streamBuilderUserWishList(int inventoryCount) {
  return StreamBuilder(
      stream: FirebaseFirestore.instance.doc("userShoppingList/${FirebaseAuth.instance.currentUser!.uid}").snapshots(),
      builder: (context, AsyncSnapshot snapshot){
        if(snapshot.hasData){
          Map<String, dynamic> userWishListData = snapshot.data.data();
          Map<String, dynamic> userAllStoresWishList = userWishListData["storeWishList"];
          if(userAllStoresWishList.keys.contains(widget.storeID)){
            //  Store in wishlist exists
            Map<String, dynamic> userStoreWishList = userAllStoresWishList[widget.storeID];
            if(userStoreWishList.keys.contains(widget.item)){
              //  Item exists in wishlist
              int wishListCount =  userStoreWishList[widget.item] as int;
              return body(inventoryCount, wishListCount);
            }
          }
          //  return something here
          return body(inventoryCount, 0);
        } else if(snapshot.hasError){
          return const Text(...);  // Text
        }
        return const Text(...);  // Text
      }
  ); // StreamBuilder
}
```

Figure 5. Wishlist StreamBuilder Code Example

4. EXPERIMENT

Central to this application is the functionality of automatically recognizing item quantities for photos that users take. As a result, testing the model's efficacy in different types of photos, including varying scales and items, is essential.

4.1. Experiment 1: Evaluating the Accuracy of the Computer Vision Model

The first experiment was conducted to gauge the accuracy of the computer vision model. A high model accuracy ensures that users can share factual information efficiently with their local communities. In our application, the accuracy concerns identifying not only what an item is but also the quantity of an item. To test its item identification capability, the model counted items in a set of validation and testing images after training in more than 200 epochs. The four metrics – precision, mAP, recall, and GIOU – were then calculated for each epoch. Besides, a qualitative investigation of counting was done by manually checking the model's output counts for several shelf images.

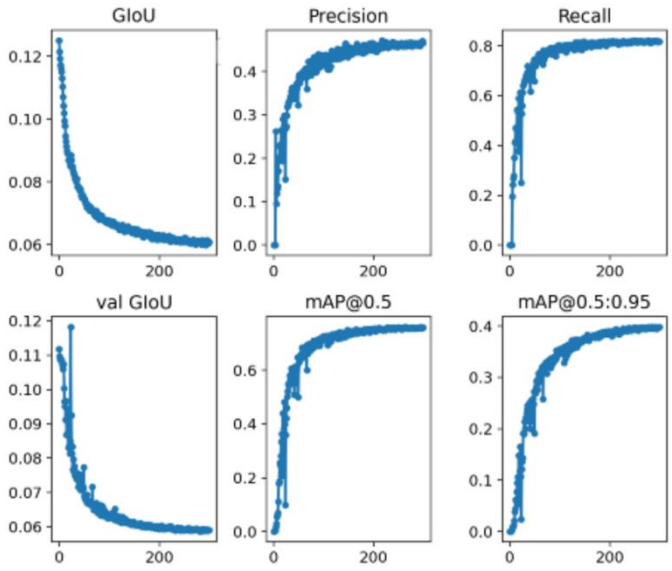

Figure 6. Model Results in Validation and Test sets

All the graphs in Figure 6 have the horizontal axis as the model's epochs and the vertical axis as the percentage. The shortcoming of the model is that its precision, which is around 50%, is slightly low, especially at a 95% Intersection Over Union (IOU) threshold, where the mAP maxed at 40%. Yet, at a 50% IOU, which indicates a decent bounding of items, the model's mAP is near 70%, a fairly high accuracy. Meanwhile, the model's recall is highly reliable; it is identified to operate on 80% of the grocery items it is trained with. Furthermore, the model localizes items well. As reflected by a 6% GIOU, which is a measure of bounding box loss, the model mostly makes correct bounds for items to identify.

In terms of counting, although the model did not count correctly in most angled photos, which are usually small-scale and of varying resolution as a result, these photos are not in consideration since the application is designed to have users take photos straight in front of the specific item of interest.

Figure 7. Example Test Images for Object Counting

Figure 7 contains two examples of large scale, well-angled, and high-resolution photos of supermarket stocks. For the photo of apples, which is an idealized, single-layered stock photo, the model correctly returns 24 apples. On the other hand, for the photo of drinks, which is a more realistic, depth-involved stock photo, the model returned 20 drinks. Even though there are clearly more drinks on the shelf, the model outputs a correct number for the first layer of the drinks, in which case the users can adjust the number.

With a high recall, the model eliminates users' need to type an item's name when updating inventory, except when the model mislabels an item, in which case the user can correct it in the review page. Similarly, the model's accurate counts in well-taken photos like ones shown above save users time with precise information. Yet, in case of miscounts due to negligence of shelf depth (illustrated in the drink example) or unclarity of a taken photo, which could be in practice, the application allows users to change the model output. Overall, the computer vision model offers helpful foundation data for users to work with, and the convenience ultimately translates to high productivity when users modify items' inventory status to be a reference for informed food purchase.

4.2. Experiment 2: Evaluating the Speed of the Computer Vision Model

The second experiment was conducted to measure the speed of the computer vision functionality. Being able to return an item count at a quick speed ensures efficiency and user experience with the application. To do this experiment, I implemented Python's timeit module, which measures the execution time of a code snippet. In an IDE, the model ran on 60 images of common grocery store items in different quantities, and the model's execution time on each image was measured. The average of all the measurements is calculated and used to represent the model's speed.

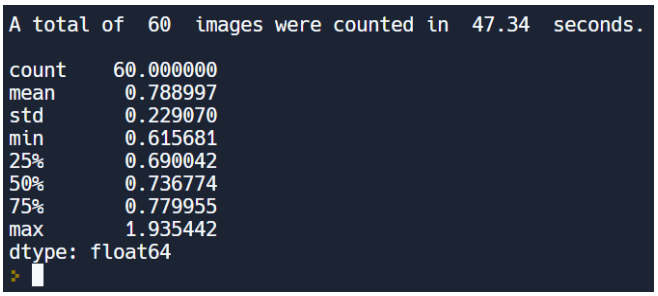

Figure 8. Timeit Console Output

As shown in the console output in Figure 8, the computer vision model finished counting all sixty images relatively quickly, averaging to less than 0.8 seconds per photo of items. Despite the fact that the longest time that the model has spent to count items in a photo is near 2 seconds, the model's counting speed is still considered to be highly consistent because of the small standard deviation, 0.23 seconds per photo of items, in counting times. Since the photos that the model is tested with are of various layouts, the model will most likely perform in a similar manner for all kinds of photos of items. Thus, the model is expected to generally finish counting a photo of food items at around 0.79 seconds.

Based on the result of this experiment, which indicates that users can receive a count of items immediately, it is reasonable to conclude that the application is easy to use and, therefore, acceptable among users. The model's capability to perform quickly fits people's fast-paced, modern lifestyle and facilitates people's effort in combating food waste.

5. RELATED WORK

Christensen, B. et al developed an application to limit food waste through charitable sales [13]. The application, Too Good to go, allows users to be informed of the food surplus in nearby grocery stores or restaurants, who provide the information on the platform. Users can then reserve the food at a discounted price for pickup at designated locations. This system effectively allows the public to make use of the potentially wasted food. Compared to this application, our

application augments the feature by specifically indicating what items the users can claim, as detected from pictures. This difference enables the users of our application to make purchasing decisions more easily.

Trax, a Singaporean company, has also developed an inventory tracking system using computer vision [14]. The cameras on shelves and ceilings hourly record the stock status, which are then analyzed by a cloud. After checking for the completeness of the photo, the computer vision system ratifies the photo quality and obtains a full picture of a shelf through panoramic stitching. It then detects the inventory's stock status, which consists of item number count, item placement, and pricing information. While this application's AI mechanism is much more robust and provides more retailing information, our application allows users to more easily retain the information they need. Users can look for the most recent photo of the stock of their desired items by selecting them in the app. This allows our application to be a handy tool for users to plan their shopping by determining where to buy an item.

Varghese, C. et al built a community between food suppliers and consumers during COVID-19 [15]. To alleviate the increasing food shortage during the pandemic, the application is a platform on which donors can post their donation information for people wanting food to pick up, and others' demand can be entered for donors to see. It fits into AI for Smart Living via Human Computer Interaction and ubiquitous computing. Our application shares the property of making use of otherwise wasted food and furthers it by easing the entire process. Suppliers do not have to specify where and when to pick up the products as the interface is set up by stores, and consumers are assumed to buy the products as soon as possible. The food quality is also guaranteed since the food comes directly from grocery stores instead of individual donors.

6. CONCLUSIONS

To combat food waste in supermarkets, this project proposes a digital, real-time supermarket inventory management system that uses computer vision. Both store managers and the public can inform others about where and how much of a food item there is by taking a photo of the items. The computer vision model, which is built in Python, counts the number of an item and returns the result to both the database Google Firebase and application UI, which is built using Flutter, to update the information. With a wishlist that documents users' personal demands, the application promotes smart buying decisions made based on trends in a user's wishlist.

Our solution relies on the data that users provide and share on the application, and the only way that users can do so is taking a photo and, if necessary, modifying the count that the built-in computer vision returns. Therefore, the computer vision model, a pivotal functionality, was experimented with, mainly through testing its accuracy and speed of counting items in photos, the two most tangible features. To test the accuracy, the model performed object detection on validation and test images and object counting on several picked images. The near 50% precision and 80% recall were accommodated by editable item names in the application. The model correctly returned single-layered counts for photos of shelf items. Meanwhile, the speed of the model is measured by its average time of counting the grocery items in sixty photos, which is 0.79 seconds per photo. Altogether, by promptly returning stock information that users can adjust and utilize, the computer vision model enhances the effectiveness and usability of the application, easing and encouraging the public's initiatives in informed food consumption.

Yet, the effectiveness of the solution could be further improved in several aspects. To start with, the computer vision model vocabulary is currently limited. While the model could explicitly identify several commonly seen items, including apples and bottled goods, it cannot perform an item count on others that the model does not recognize. The model either ignores the items or

miscounts with irrelevant items. In addition, the accuracy of the model could definitely undergo more optimizations. Currently, the model can correctly count items that are photographed well-angled. However, it is unrealistic that users can take photos upright every time as that would require too much effort.

To address these limitations, expanding the scope of training data will improve the model's capacity. The model can then identify and perform counting on the objects. In addition, experimenting with parameters of the model can help find the most accurate specs. Along the way, providing the model with more data can prevent the model from obtaining skewed results.

REFERENCES

[1] O'Shea, Keiron, and Ryan Nash. "An introduction to convolutional neural networks." arXiv preprint arXiv:1511.08458 (2015).

[2] Zeide, Anna. "Grocery garbage: food waste and the rise of supermarkets in the mid-twentieth century United States." History of Retailing and Consumption 5.1 (2019): 71-86.

[3] Eriksson, Mattias, and Johanna Spångberg. "Carbon footprint and energy use of food waste management options for fresh fruit and vegetables from supermarkets." Waste Management 60 (2017): 786-799.

[4] Aschemann-Witzel, Jessica, et al. "Consumer-related food waste: Causes and potential for action." Sustainability 7.6 (2015): 6457-6477.

[5] Curry, Nathan, and Pragasen Pillay. "Biogas prediction and design of a food waste to energy system for the urban environment." Renewable Energy 41 (2012): 200-209.

[6] Poyatos-Racionero, Elisa, et al. "Recent advances on intelligent packaging as tools to reduce food waste." Journal of cleaner production 172 (2018): 3398-3409.

[7] Kor, Yasemin Y., Jaideep Prabhu, and Mark Esposito. "How large food retailers can help solve the food waste crisis." Harvard Business Review 19 (2017).

[8] Chaudhary, Sanjay, and P. K. Suri. "Agri-tech: experiential learning from the Agri-tech growth leaders." Technology Analysis & Strategic Management (2022): 1-14.

[9] Skaggs, Richard L., et al. "Waste-to-Energy biofuel production potential for selected feedstocks in the conterminous United States." Renewable and Sustainable Energy Reviews 82 (2018): 2640-2651.

[10] Ehrlen, Johan. "Why do plants produce surplus flowers? A reserve-ovary model." The American Naturalist 138.4 (1991): 918-933.

[11] Consumer Reports. "What to do when there are too many product choices on the store shelves?." Consumer Reports (2014).

[12] Islam, Md Olioul. "A high embedding capacity image steganography using stream builder and parity checker." 2012 15th International conference on computer and information technology (ICCIT). IEEE, 2012.

[13] Baglioni, Simone, Benedetta De Pieri, and Tatiana Tallarico. "Surplus food recovery and food aid: The pivotal role of non-profit organisations. Insights from Italy and Germany." VOLUNTAS: International Journal of Voluntary and Nonprofit Organizations 28.5 (2017): 2032-2052.

[14] Coifman, Benjamin, et al. "A real-time computer vision system for vehicle tracking and traffic surveillance." Transportation Research Part C: Emerging Technologies 6.4 (1998): 271-288.

[15] Varghese, Christina, Drashti Pathak, and Aparna S. Varde. "SeVa: a food donation app for smart living." 2021 IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC). IEEE, 2021.