T.R.

GEBZE TECHNICAL UNIVERSITY

FACULTY OF ENGINEERING

DEPARTMENT OF COMPUTER ENGINEERING

SMART MOBILE APPLICATION FOR IDENTIFICATION AND TRACKING PACKAGING IN RECYCLING

EMRE GÜVEN

SUPERVISOR
PROF. DR. YUSUF SINAN AKGÜL

GEBZE
2024

T.R.
GEBZE TECHNICAL UNIVERSITY
FACULTY OF ENGINEERING
COMPUTER ENGINEERING DEPARTMENT

SMART MOBILE APPLICATION FOR IDENTIFICATION AND TRACKING PACKAGING IN RECYCLING

EMRE GÜVEN

SUPERVISOR
PROF. DR. YUSUF SINAN AKGÜL

2024
GEBZE

This study has been accepted as an Undergraduate Graduation Project in the Department of Computer Engineering in 25/01/2024 by the following jury.

JURY

Member
(Supervisor) : Prof. Dr. Yusuf Sinan Akgül

Member : Dr. Yakup Genç

ABSTRACT

This project aims to revolutionize waste management in recycling by developing an object recognition model capable of identifying packaging waste and waste containers through live video analysis.

The chosen MobileNetV2 SSD FPN-Lite model, optimized for devices with limited performance, undergoes training with a diverse dataset including plastic, metal, and glass packaging waste, as well as waste containers. Following training, the model is converted to a Tensorflow Lite-compatible format for seamless integration with mobile devices.

On the Android application development, the live video stream from the camera module is processed to provide discrete input to the model, and sequentially recognized objects are presented as output. Real-time object recognition is executed through continuous analysis of these outputs. The recognition process initiates upon detecting the waste container, followed by an attempt to identify the act of throwing packaging waste into the waste container.

The user can view the locations of recycling waste containers within the application and contribute to recycling by throwing waste packaging to appropriate places. In addition, the waste packaging recognized in the throwing waste process are shown to the user at the end of the process and feedback is provided. Contributions to recycling are accessible for user review, in detail as a list and pie chart.

Keywords: recycling, waste packaging, object recognition, Android application, SSD MobileNetV2, TF Lite.

ACKNOWLEDGEMENT

I would like to thank to my advisor Prof. Dr. Yusuf Sinan Akgül and to my industrial advisor, Computer Engineer Yusuf Demir, the developer of the Biriktir application, for their valuable guidance and support throughout this project.

Emre Güven

LIST OF SYMBOLS AND ABBREVIATIONS

Symbol / Abbreviation	**Explanation**
ML	Machine Learning
TF Lite	TensorFlow Lite
SSD	Single Shot Multibox Detector
FPN-Lite	Feature Pyramid Network Lite
FPS	Frames per second
mAP	Mean average precision
mAP50	Mean average precision calculated at an intersection over union (IoU) threshold of 0.50, representing model accuracy for ”easy” detections.
mAP75	Mean average precision calculated at an intersection over union (IoU) threshold of 0.75, representing model accuracy for ”moderate” detections.
mAP50-95	The average of mean average precision computed at varying IoU thresholds ranging from 0.50 to 0.95, providing a comprehensive assessment of model performance across different levels of detection difficulty.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

In this project, I developed an object recognition model utilizing the MobileNetV2 SSD FPN-Lite architecture to identify packaging waste and waste containers. Additionally, through the implementation of a mobile application, I applied the model to real-time video processing, enabling the recognition of instances where packaging waste is being thrown into the appropriate waste container.

1.1. Project Description

The application enables users who wish to contribute to recycling to find nearby recycling containers for proper throwing the packaging waste into the container. Following the verification of the container at their selected location, users can utilize their smartphones to capture the process of throwing the packaging waste into the container. The application facilitates the recognition of the waste thrown into the container, allowing users to save their contribution and subsequently track it in detail.

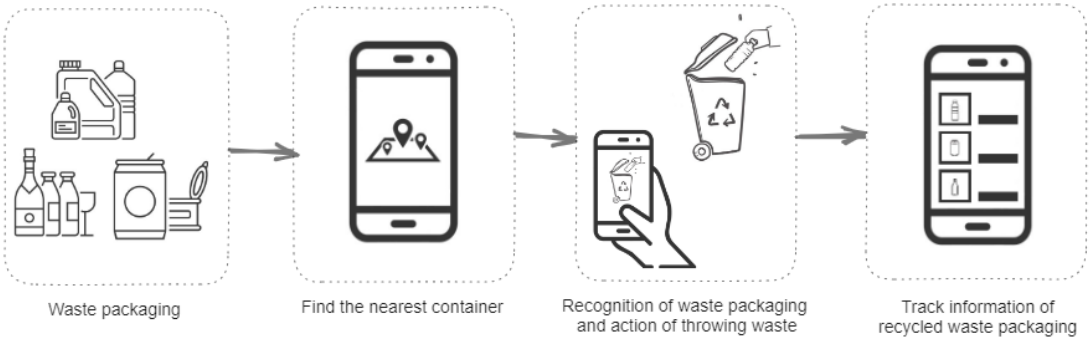


Figure 1.1: Application Fundamental Design

1.2. Success Criteria

- The type and the quantity of waste packaging materials should be detected with at least 80% accuracy.
- Containers should be detected with at least 90% accuracy.
- At least 5 frame per second should be received and processed.

2. OBJECT RECOGNITION MODEL

An object recognition model needs to be developed for the identification of packaging waste and waste containers. A base model must be chosen and trained on a diverse dataset containing images of plastic, metal, glass packaging waste materials, and waste containers. Afterwards, the trained model should be converted to a TF Lite model to run on an Android application [1] [2].

2.1. Chosen Model

The chosen base model should operate efficiently on mobile devices with limited performance, ensuring both speed and satisfactory accuracy. Additionally, the model should provide output indicating the locations of recognized objects in the image. The preferred model meeting these criteria is MobileNetV2 SSD FPN-Lite from TF Lite Object Detection Models [3], an advanced version of MobileNetV2. This model employs the SSD algorithm as its recognition network and adopts FPN-Lite as the feature extractor.

2.2. Dataset

To prepare a dataset, images containing plastic, metal, glass packaging waste materials, and waste containers should be found and labeled with bounding boxes, by using Label Studio. Videos containing packaging waste being thrown into waste containers, test videos of the Biriktir [4] application, were processed, resulting in 3421 images containing packaging waste and waste containers. These images were labeled with 13 separate labels using Label Studio [5]. Labels 2.1 and an example annotation 2.1 is presented.

Table 2.1: Label Informations

Label	Type	Number of Labelling
Atık Kutusu	-	2040
Soda Şişesi	Glass	920
Alkol Şişesi	Glass	330
Cam Kavanoz	Glass	241
Plastik Pet Şişe	Plastic	803
Plastik Gazlı Şişe	Plastic	327
Plastik Mat Şişe	Plastic	292
Plastik Bardak	Plastic	109
Metal İçecek Kutusu	Metal	632
Konserve Kutusu	Metal	187
Metal Alüminyum	Metal	165
Kavanoz Kapağı	Metal	158
Metal Deodorant	Metal	51

Figure 2.1: Example Annotation

2.3. Model Training

The chosen base model MobileNetV2 SSD FPN-Lite has been pre-trained on the COCO 2017 dataset using images scaled to a resolution of 320x320, achieves 22.2 mAP on COCO17 Val [6]. The model is trained on the 80% of packaging dataset with 10% validation dataset. The model training process can be examined with the values in the graphs, in Figures 2.2, 2.3, 2.4, 2.5, 2.6.

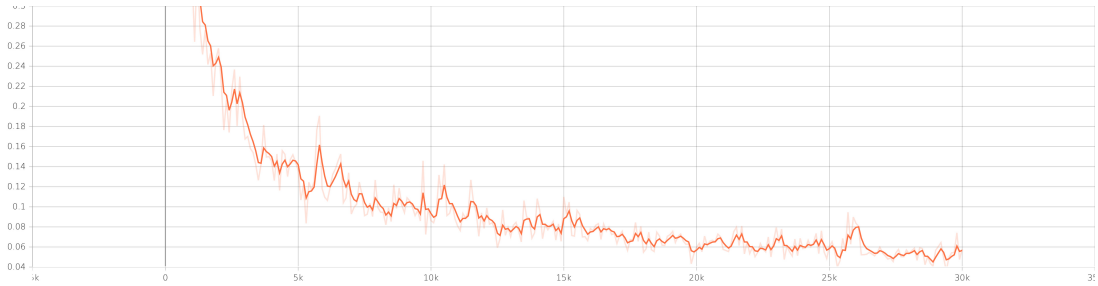


Figure 2.2: Loss/Classification Loss

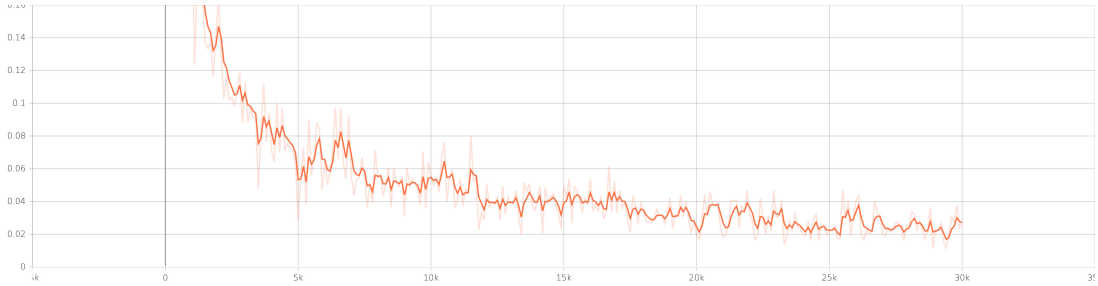


Figure 2.3: Loss/Localization Loss

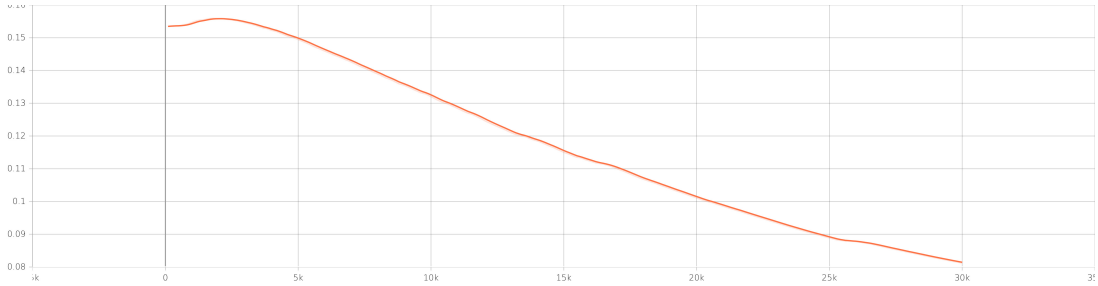


Figure 2.4: Loss/Regularization Loss

Figure 2.5: Loss/Total Loss

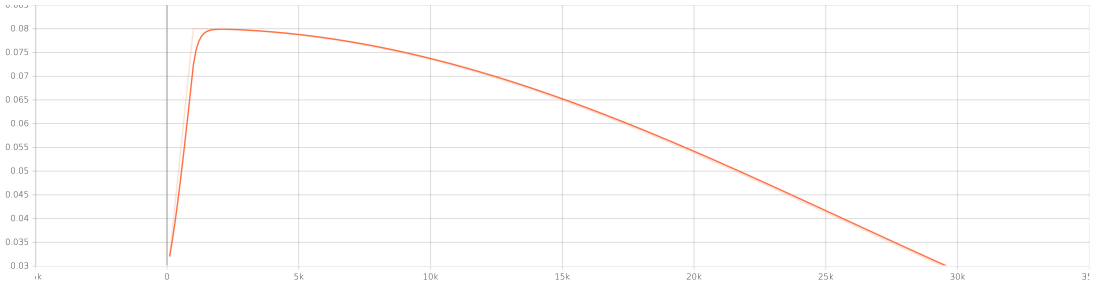


Figure 2.6: Learning Rate

2.3.1. Test Results

Tests were performed with randomly selected 10% data of the entire dataset. Results can be seen in Tables 2.2, 2.3, 2.4.

Table 2.2: mAP50

Object Class	**mAP (%) @ 0.5**
Alkol Şişesi	85.26
Atık Kutusu	96.49
Cam Kavanoz	93.36
Kavanoz Kapağı	92.31
Konserve Kutusu	72.41
Metal Alüminyum	76.19
Metal Deodorant	30.48
Metal İçecek Kutusu	72.88
Plastik Bardak	99.22
Plastik Gazlı Şişe	66.54
Plastik Mat Şişe	95.96
Plastik Pet Şişe	82.08
Soda Şişesi	84.82
mAP	80.62

Table 2.3: mAP75

Object Class	**mAP (%) @ 0.75**
Alkol Şişesi	70.13
Atık Kutusu	89.31
Cam Kavanoz	87.50
Kavanoz Kapağı	92.31
Konserve Kutusu	58.92
Metal Alüminyum	63.81
Metal Deodorant	10.00
Metal İçecek Kutusu	65.37
Plastik Bardak	84.76
Plastik Gazlı Şişe	62.76
Plastik Mat Şişe	95.96
Plastik Pet Şişe	66.65
Soda Şişesi	62.00
mAP	69.96

Table 2.4: mAP50-95

Class	**Average mAP (%) @ 0.5:0.95**
Atık Kutusu	75.82
Cam Kavanoz	67.86
Soda Şişesi	57.73
Alkol Şişesi	58.42
Plastik Gazlı Şişe	47.16
Plastik Pet Şişe	57.61
Plastik Bardak	75.09
Plastik Mat Şişe	77.36
Metal İçecek Kutusu	54.70
Metal Alüminyum	56.56
Metal Deodorant	14.14
Konserve Kutusu	52.04
Kavanoz Kapağı	65.76
Overall mAP	58.48

3. APPLICATION DEVELOPMENT

3.1. Technologies

In this section, the technologies required for the application development are listed.

- Kotlin: Mobile application is developed as a native Android application with Kotlin programming language.
- TF Lite Plugin: This plugin enables to run ML models in constrained performance devices.
- Google Maps Plugin: This plugin allows displaying a map in the application. It allows integration with the Native Google Maps application.

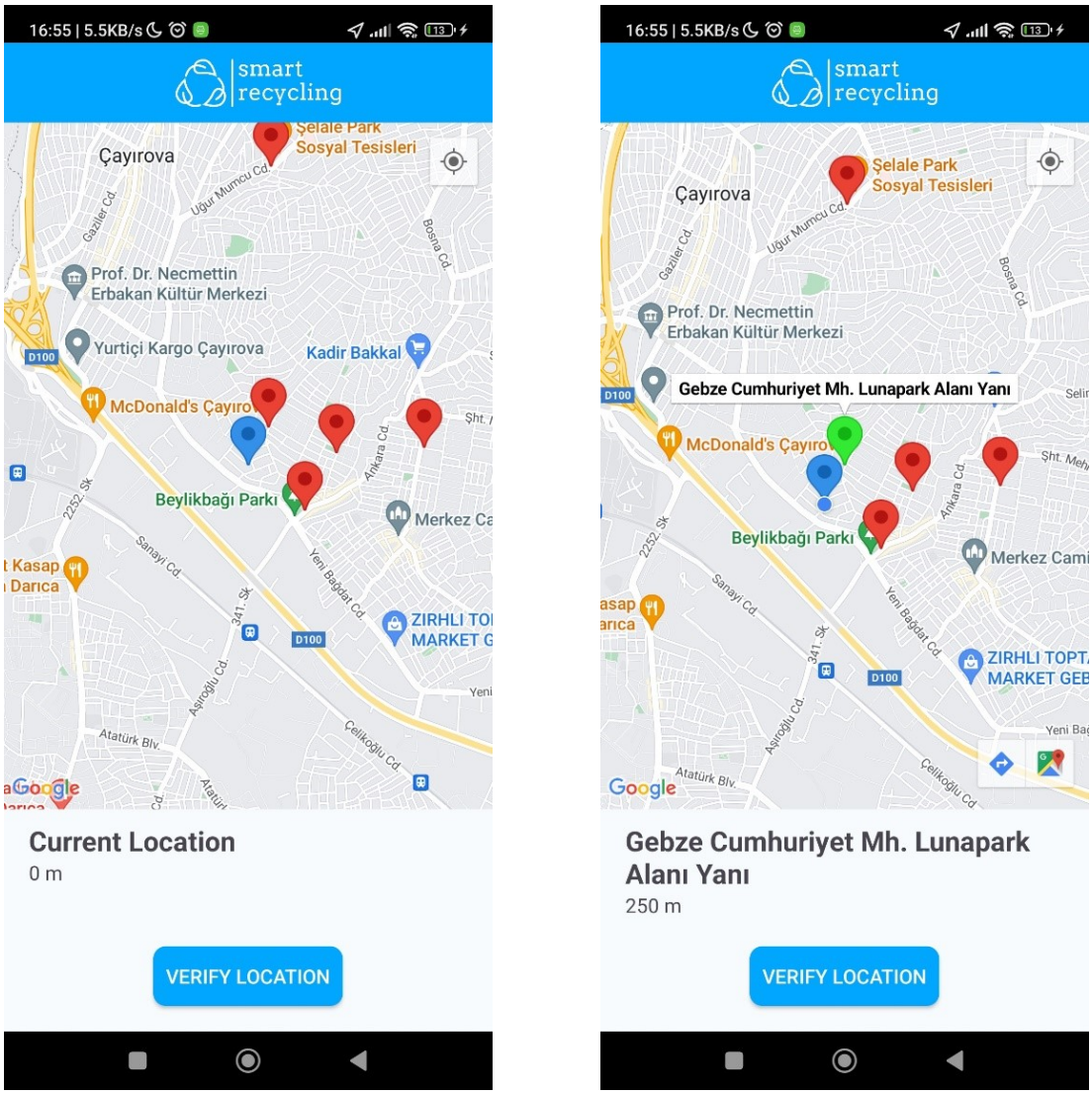
3.2. Screen UI Designs and Functionalities

This section explains the functionalities of application for each screen.

Figure 3.1: Home Screen

3.2.1. Location Selection-Verification Screen

By using map support with Google Maps API [7] and the waste collection container data set among the Kocaeli Metropolitan Municipality Data Sets [8], waste collection containers are displayed according to their distance to the users' current locations.

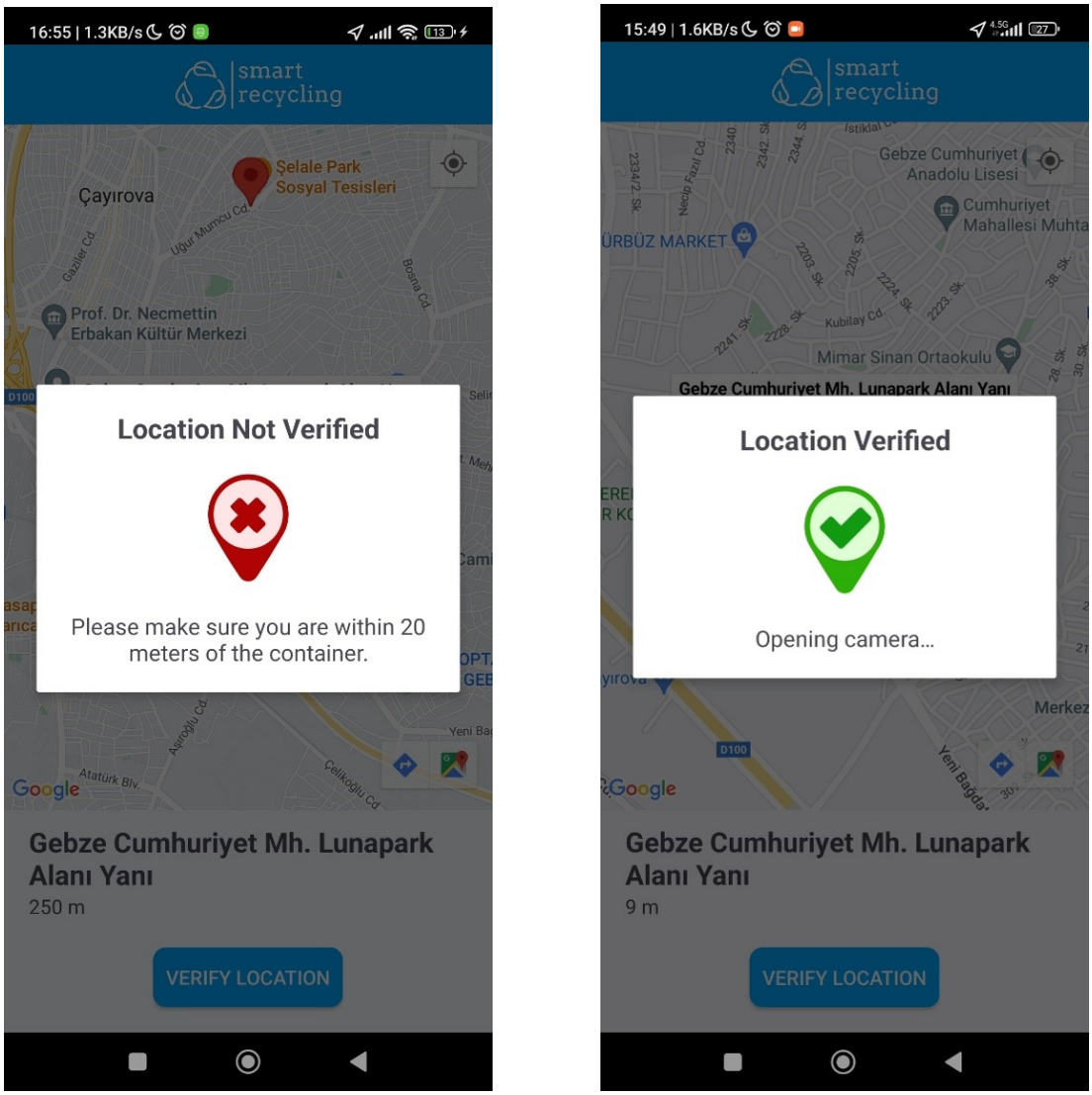


(a) Initial map

(b) Select a location

Figure 3.2: Nearest Container Locations on Map

Container verification is performed to enable users to throw of waste into recycling containers. Users must be within approximately 20m of waste containers to ensure verification.



(a) Not verified

(b) Verified

Figure 3.3: Location Verification

3.2.2. Detection Screen

The camera should be kept in a steady position throughout the entire process. If there is movement, the user is warned and the process is suspended. The movement of the camera is controlled by the accelerometer sensor of the device.

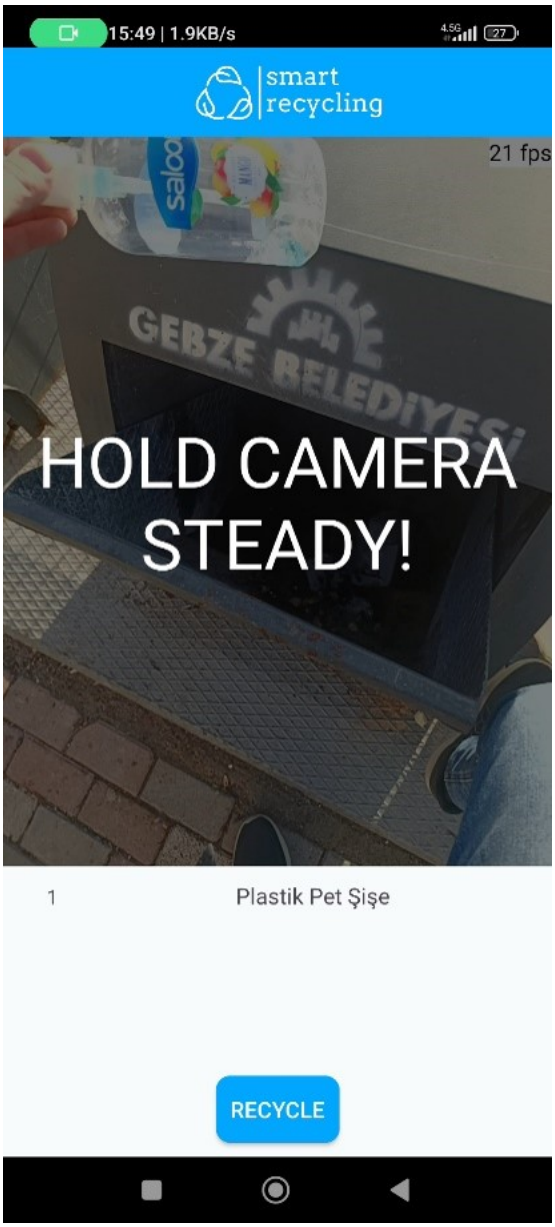
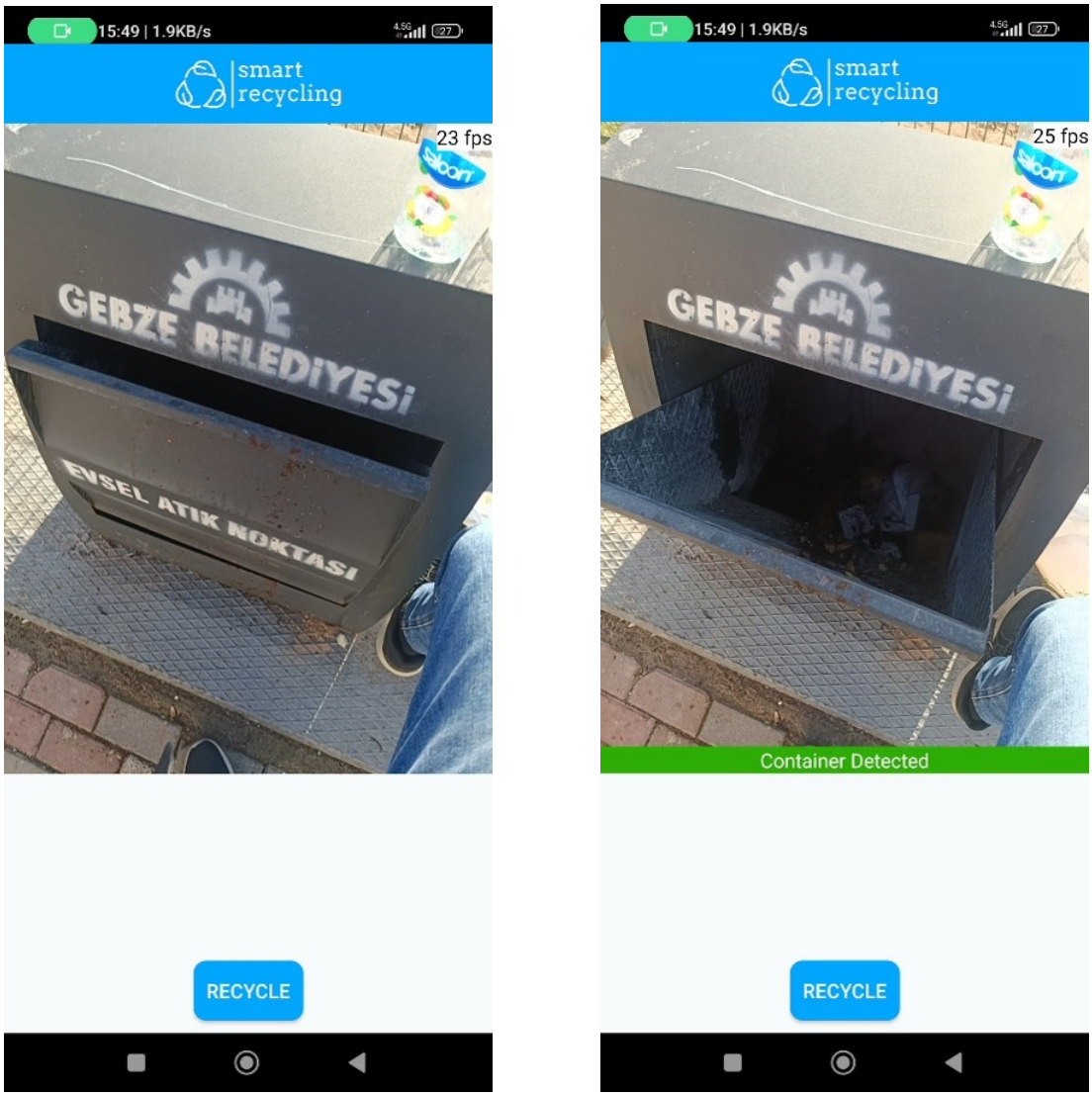


Figure 3.4: Holding Camera Steady

When the camera is stable, the ML model initiates its operation. The model promptly captures an image from the video, analyzes it, and provides outputs identifying the objects present. This continuous process illustrates the methodology behind achieving real-time recognition.

The initiation of the recognition process depends on the recognition of the entrance of the container . The user receives a message once the container is successfully recognized .3.5(b).

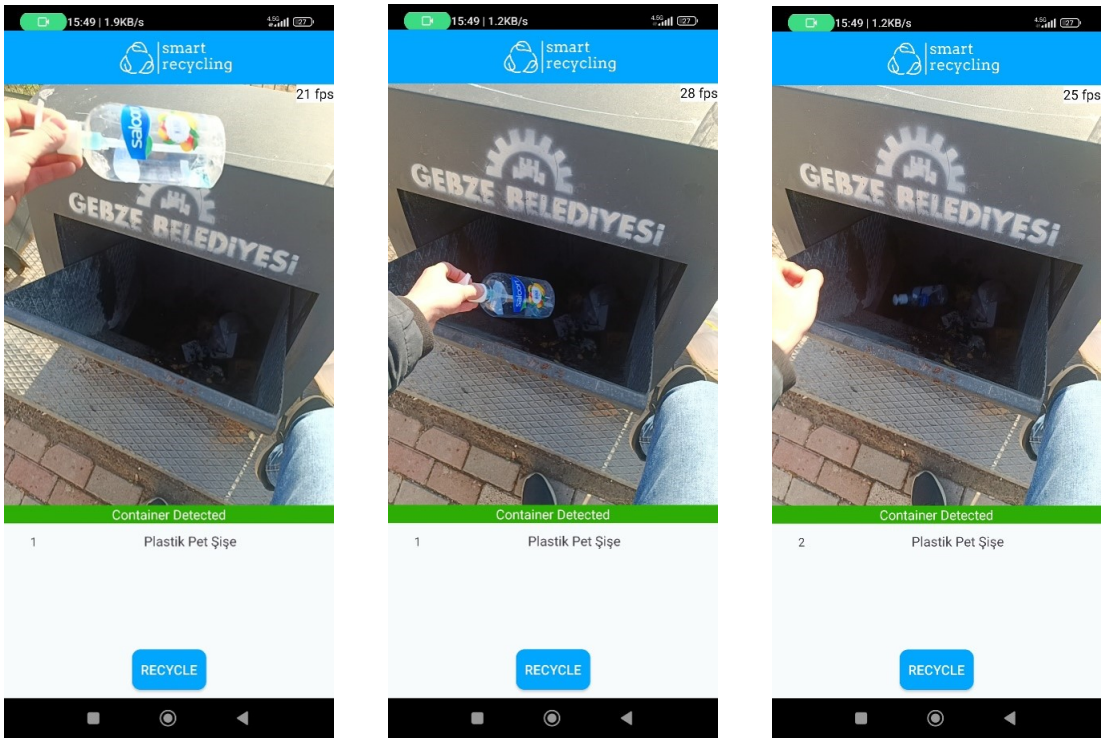


(a) Container entrance is not recognized

(b) Container entrance is recognized

Figure 3.5: Container Recognition

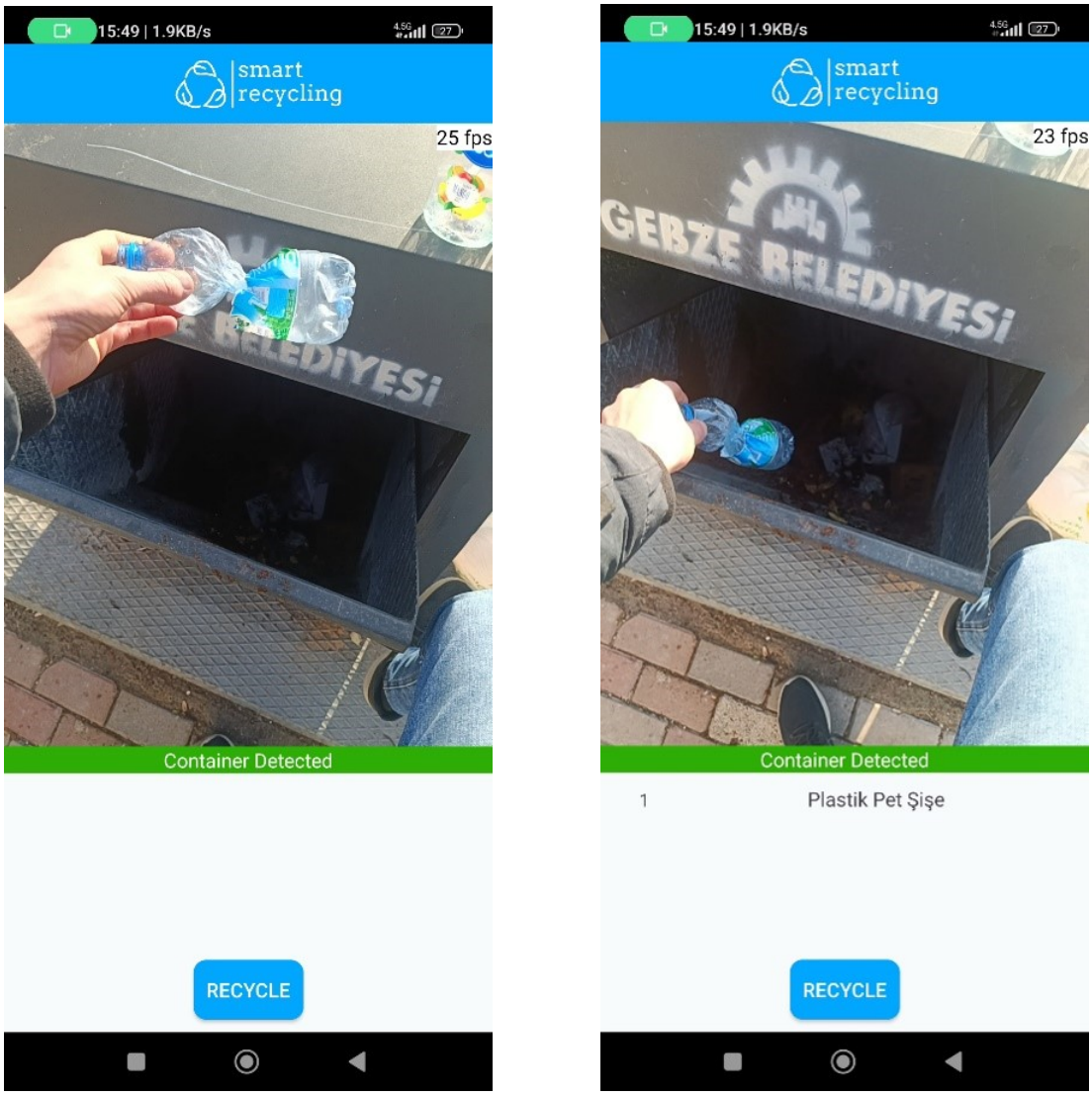
The user shows the packaging waste to the camera outside the entrance of the container, shown in Figure 3.6 (a), allowing the packaging waste to be recognized by the object recognition model. In the subsequent process, the application constantly examines the instantaneous position information of the packaging waste, given in the outputs of the object recognition model. When the packaging reaches the waste container entrance, it is marked as ready-to-throw waste, shown in Figure 3.6 (b). If the packaging waste disappears by moving into the container, it is determined to be thrown into the waste container, shown in Figure 3.6 (c).



(a) Detect packaging waste (b) Ready to throwing (c) Recognize the action

Figure 3.6: Throwing Action Recognition Example 1

In Figure 3.7, another process for recognizing throwing action is presented.

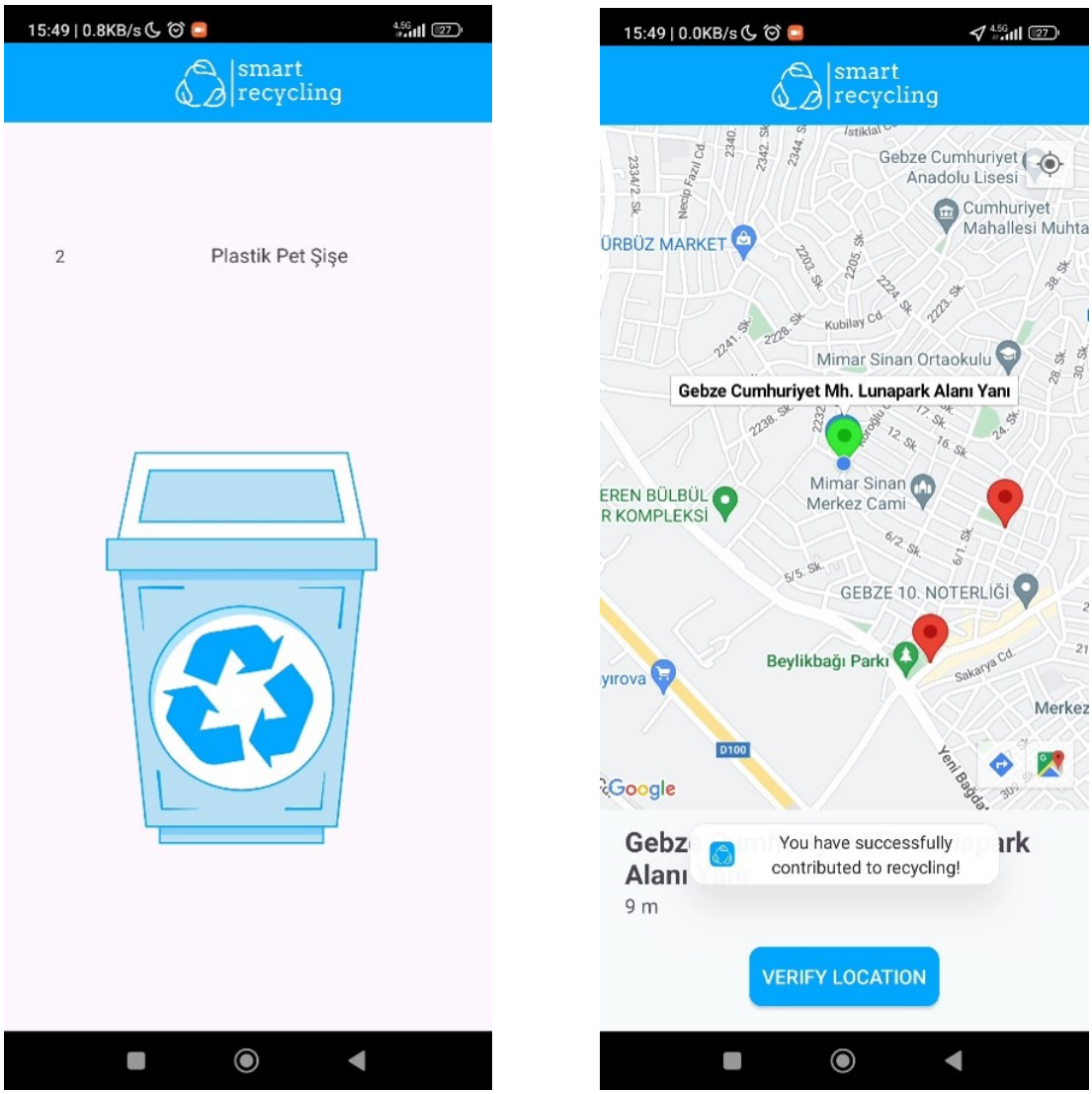


(a) Detect packaging waste

(b) Recognize the action

Figure 3.7: Throwing Action Recognition Example 2

The application provides users with feedback on their contributions to recycling, 2 different feedback styles are shown in the Figure 3.8.

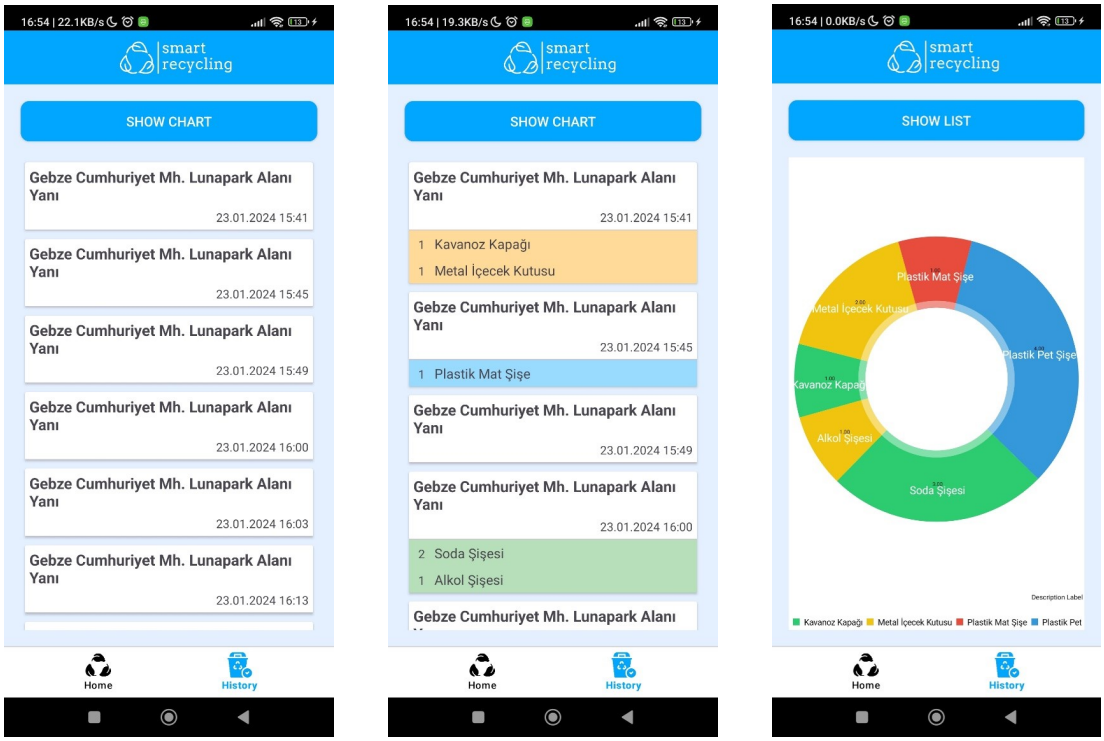


(a) (b)

Figure 3.8: Feedback Styles

3.2.3. Recycling History Screen

On this screen, users have the opportunity to track their contributions organized by date and categorized by waste container information, as seen in Figure3.9(a). They can also view details about the packaging waste thrown during each recycling process, shown in Figure3.9(b). In Figure3.9(c), a pie chart, showing the distribution of total waste thrown, is displayed.



(a) List view of recyclings sorted by date

(b) List view of recyclings with detail

(c) Piechart representation of recyclings

Figure 3.9: Recycling History View Options

3.3. Action Recognition Method

This section explains the fundamental working of the action recognition algorithm. The algorithmic explanation is displayed in 1.

Algorithm 1 Continuous Monitoring and Packaging Recognition

```
1:  Continuous Monitoring:
2:  while camera is active do
3:      if camera is steady then
4:          Make an object recognition.
5:      end if
6:
7:      Container Detection:
8:      while container is not detected do
9:          Wait for the detection of a container.
10:     end while
11:
12:     if container is detected then
13:         Packaging Position Check:
14:         for each detected packaging do
15:             if packaging is ready for throwing at previous detection then
16:                 if packaging is not inside the container entrance at its first seen
    position then
17:                     Wait until the packaging disappears from the view.
18:                     Mark the packaging as thrown into the container.
19:                 end if
20:             end if
21:         end for
22:
23:         Packaging Readiness Check:
24:         for each detected packaging do
25:             if packaging is currently inside the container entrance then
26:                 Mark it as ready for throwing.
27:             else
28:                 Mark it as not ready for throwing.
29:             end if
30:         end for
31:     end if
32: end while=0
```

4. CONCLUSIONS

In conclusion, this project aims to revolutionize recycling waste management by developing an object recognition model using the MobileNetV2 SSD FPN-Lite. The model, trained on a diverse dataset, recognizes packaging waste and waste containers. The Android app processes the live video stream for real-time object recognition, starting the process upon detection of a waste container and attempting to identify the act of throwing packaging waste. Recognition of the act of throwing waste involves analyzing the position and visibility changes of recognized objects in the current image extraction. Users can view recycling waste container locations, contribute to recycling, and receive feedback on recognized waste packaging.

The model, which gives a reasonable accuracy value in mAP50 and mAP75 object recognition according to success criteria, is open to improvement at high IoU values and in general accuracy value mAP50-95. Augmenting the dataset with high IoU values for packaging waste images can enhance model accuracy. The inclusion of images with fewer labels for packaging waste compared to other categories can further improve accuracy. Moreover, considering the 640x640 input size option of MobileNetV2 SSD FPN-Lite, choosing this version for higher-performance devices may lead to increased accuracy. These enhancements collectively aim to optimize the project's efficiency and accuracy in waste management and recycling.

BIBLIOGRAPHY

[1] TensorFlow. “Tensorflow lite for android.” (2024), [Online]. Available: https://www.tensorflow.org/lite/android.

[2] TensorFlow. “Tensorflow lite model metadata.” (2024), [Online]. Available: https://www.tensorflow.org/lite/models/convert/metadata.

[3] TensorFlow. “Object detection with tensorflow lite on android.” (2024), [Online]. Available: https://www.tensorflow.org/lite/android/tutorials/object_detection.

[4] Biriktir. “Biriktir - mobile application.” (2024), [Online]. Available: https://www.biriktir.app/.

[5] L. Studio. “Label studio - open source data labeling.” (2024), [Online]. Available: https://labelstud.io/.

[6] T. Models. “Ssd mobilenet v2 fpn-lite 320x320 configuration file.” (2024), [Online]. Available: https://github.com/tensorflow/models/blob/master/research/object_detection/configs/tf2/ssd_mobilenet_v2_fpnlite_320x320_coco17_tpu-8.config.

[7] G. M. Platform. “Google maps android sdk documentation.” (2024), [Online]. Available: https://developers.google.com/maps/documentation/android-sdk/overview.

[8] K. M. Municipality. “Mobile waste collection centers dataset.” (2024), [Online]. Available: https://veri.kocaeli.bel.tr/dataset/mobil-atik-getirme-merkezleri.

APPENDICES

GitHub Repository

You can find the application source code and model training code on GitHub: https://github.com/eemreguven/CSE495-GraduationProject

YouTube Demo

You can view a trailer of the project on YouTube: https://www.youtube.com/watch?v=kg9qFErldlU