Recycling Assistant

Identifying and Categorizing Objects based on Recyclability

Doo Woong Chung
Dept. Information Systems
Hanyang University
Seoul, South Korea
dwchung@hanyang.ac.kr

Kim Soohyun
Dept. Information Systems
Hanyang University
Seoul, South Korea
soowithwoo@gmail.com

Lim Hongrok
Dept. Information Systems
Hanyang University
Seoul, South Korea
hongrr123@gmail.com

***Abstract*—Correctly identifying and categorizing Recyclables can be difficult at times. As such, the Recycling Assistant hopes to be able to both educate, as well as, assist in the matter of correctly identifying which category of recycling an item belongs to, as well as determine if an item cannot be recycled in its current state (and belongs in the general waste).**

***Index Terms*—identification, detection, classification, opencv, recycling, waste**

I. Introduction

The Recycling Assistant is a project hoping to identify, and categorize recyclable materials. In this form, we envision the Recycling Assistant to allow the user to hold up an item and display which category of recycling the item belongs in. In the event that the item is not recyclable, it will notify the user accordingly.

Recycling can be convoluted at times, with people sometimes confusing what is recyclable and what is not. As a result, objects are often incorrectly disposed of, with either recyclables ending up in general waste, or objects that aren't recyclable, heading to the recycling plant. This leads to only a fraction of the submitted recycled waste actually being able to be recycled, with the remainder heading back to landfills.

Our goal is to provide an assistant program that can quickly guide the user to which type of category of recycling the item that the user shows the camera, belongs in - if it is recyclable to begin with. Otherwise, they will be notified that it is more fitted to general waste.

As AI usage in environmentally friendly initiatives is something that is of high interest, waste-related datasets are plentiful. We hope to leverage these datasets in order to train the neural network, while OpenCV will be used to receive and output a processed video stream. A database may also be maintained in order to provide statistics on which types of objects are most frequently identified, which can be linked to by other applications or services.

As a stretch goal, we hope to be able to support the detection of different materials in a way such that the Assistant can differentiate between several types of material on the same object, and inform the user on why the object is not recyclable in its current state (ie. Plastic film on a paper box needs to be separated). We envision the final product to be largely focused on the real-time detection and classification of objects such that it can differentiate between recyclable categories, as well as differentiate between recyclables and non-recyclables.

II. Requirements Analysis

A. User Interface

The application should have a clear interface in which the user may hold up the object in question to the camera, and the application should be able to output what type of object it is via, and which category of recycling it belongs in text output, as well as show which item it is referring to through graphics such as bounding boxes.

B. Input Support

The application should be able to support a real-time input of a camera, through either a physically connected input, or a local network camera input.

C. Output Feedback

The application should be able to draw at the very least, bounding boxes, or correctly segment the object, and output it on the display. This is in order to output appropriate feedback to the user on which item was identified, and how it came to the conclusion it did. Then, it should output the result through text, as well as the confidence level.

D. Real Time Analysis

The application should be able to handle real-time detection, as well as be able to analyze and classify the object based on the input, and visual output (ie. text and bounding boxes).

E. Database and Statistics

The application should store statistics on the types of objects that were displayed to the camera, to a database. This database may be used in the future in order to draw and show statistics on for example, the most frequently inquired object, and object classification.

F. Extra - Focused Feedback

As a stretch goal, the application should be able to add a more specific explanation to each output, educating the user on why a certain object is not recyclable, or why it is not recyclable in its current state.

III. Existing Products

There are quite a bit of related existing products for this Recycling related AI work, as it is a field of interest in both the AI sector, as well as having a positive environmental affect.

A. Recycle Mate [1]

An Android/iOS application which scans an item, and identifies which bin it goes into. It is highly localized, directing users into the correct bin to dispose of based on information provided by the local council - thus, it is limited to residents in New South Wales only. It takes a picture of the item, analyzes it and directs the user to which color bin it needs to be disposed of in.

B. Bin-e Smart Waste Bin [2]

A "Smart Recycling Bin", which automatically sorts the inserted item into the correct bin. It uses an inserted camera alongside object recognition in order to correctly sort the item.

C. World Waste Platform [3]

The "Let's Do It AI Project" is a project developed in conjunction with the Let's Do It Foundation and SIFR in partnership with Microsoft. It detects trash, but does not classify between whether or not an item is recyclable, as it is focused on identifying litter.

D. TrashBot [4]

An industrial and commercial application of AI trash sorting, TrashBot identifies if an object inserted into the TrashBot Smart Bin is recyclable, compost-able or belongs in the landfill.

E. Greyparrot [5]

An industrial approach to waste recognition, Greyparrot uses AI to recognize and classify waste composition at an industrial level - focusing on identifying waste passing through a camera on a conveyor belt.

F. Intuitive AI [6]

A similar approach, the Intuitive AI uses computer vision and machine learning to identify the item the user is holding, and tell them which category of recycling it misses - alongside yelling at the user if they recycle incorrectly or praising the user if they do it correctly.

References

[1] "Recycle Mate." Recycle Mate, https://recyclemate.com.au/.
[2] "Bin-e Smart Waste Bin." Bin-e, https://www.bine.world/.
[3] "World Waste Platform." Let's Do It AI Project, https://ai.letsdoitworld.org/
[4] "TrashBot." CleanRobotics, https://cleanrobotics.com/trashbot/
[5] "Greyparrot AI." Greyparrot, https://www.greyparrot.ai/waste-composition-analysis-software
[6] "Trash-Talking Recycling AI." Nvidia Blogs, https://blogs.nvidia.com/blog/2020/02/03/intuitive-ai-schools-you-on-recycling/

IV. Initial Thoughts / Project Preamble

As mentioned previously, as AI in Environmental related work seems to be of high interest, there are quite a few already existing products and concepts - some of which have already been commercialized or industrialized. As a result, there are a lot of great resources that will be of great use in relation to this project.

However, there are some slight differences that can be seen from several of the projects. Of these existing projects, the one closest in terms of end-goal is the "Intuitive AI" project - minus the trash-talking aspect. The base goal is the same - the goal of advising and educating the user on how to recycle a certain object.

Our project, however, also has a stretch goal - the hope of being able to differentiate between "hybrid materials" on the same object. For example, if a paper box also has a plastic part - it needs to be separated.

Fortunately, there are a lot of available datasets that we can utilize, such as TACO, which also supports object segmentation and is full of COCO annotated images, as well as many more other datasets pertaining to waste.

Performance and overall efficiency is also of important, as if the application takes an exorbitant amount of resources in order to operate, then it is not exactly "portable". As a result, we're looking at detection systems such as YOLO due to its performance and high resulting framerate.

We'll be looking at using AWS SageMaker in order to train the model. Getting a more tuned model for more optimized inference performance would be helpful in achieving a more performant application, alongside better detection/classification performance.

As for video input, we're looking at using OpenCV as OpenCV specializes in computer vision, as well as natively supporting streaming input. OpenCV can output "blobs", which can then be fed into the model conveniently. In adttion, as OpenCV and other aforementioned resources all contain a significant amount of documentation, it would ease and help troubleshooting down the road.

An SQL database will also be used in order to store statistics and data of the items (such as classification, confidence, etc), that can later then be used by other applications, or, in the context of development, be used in order to revise and train the model for better accuracy and detection.

Our first proposal of this project was on a "Smart Recycling Bin" - an idea that has a fair amount of already existing implementations. As a result, we revised our project to being used more as a helper tool in places such as the recycling area - however, after visiting the LG showroom, we think that it could also be helpful in personal devices. As a result of this, we ended up thinking more from static image scanning to real-time object detection and analysis.

V. Role Assignment

Name	Main Role	Responsibility Description
Doo Woong Chung	Back-End	Handling of the Input/Output system - Implementation of OpenCV into the application, as well as handling the output processing such as text output/bounding boxes. Repository Maintainer - Handling Repository Commits, General Maintenance. General Application Architect - Architect of general systems used in the application, and how they'll link together to form the final product. Base Application Prototyping - Prototyping with related applications that will be used, such as OpenCV to test integration.
Kim Soohyun	User/Front-End	User Interface System - Analysis pertaining to User Experience, and User Interface. Team Lead - Monitoring overall team performance level and maintaining communication. Project Manager - Monitoring overall progress of the project as well as upcoming deadlines. Tech Blog Maintainer - Maintaining the Team's Tech Blog.
Lim Hongrok	Back-End	Handling of the Database and Model related system - Implementation of PostgreSQL into the application, as well as handling the input of data into the DBMS. Model Training, Prototyping and Evaluation - Prototyping the Model, alongside training and evaluating/testing the Model. Amazon Web Services Handler - Handles AWS related matters, such as usage and configuring of Amazon Web Services that may be used in the training of the model, such as SageMaker. QA/Performance Testing - General Quality Assurance Testing, and Performance Testing (ie. Checking whether the model correctly identifies and classifies the object at a reasonable framerate).

VI. Design & Architecture (Draft)

At its core, the application handles a real-time video input through either connected network camera, or a physically connected camera (eg. USB). Through OpenCV, the application then analyzes the input and draws a bounded box around the identified object based on its stored/frozen model. The application then returns feedback through text on whether the object is recyclable, or belongs in general waste. If the item is recyclable, it returns feedback based on which type of recycling it belongs in alongside the confidence level. In the event that the item is unknown, or does not return a valid result, the application will either direct the user to the general waste, or return feedback that the application is unsure.

The model is trained using one of the plentiful provided databases such as the TACO Dataset, as it contains a plethora of COCO formatted annotated images. A real-time detection system such as YOLO or RCNN may be leveraged in order to provide fast detection of objects. As the application needs to keep a reasonable framerate, there will need to be some experimentation between a cross of speed and detection accuracy.

Once the item is identified and classified, the timestamp, identification, classification category will be saved to the local database. Though this database will not be normally accessible to the user, it will be designed in such a way that future feature additions or applications may access it and present data.

Input	Processing	Output
Camera Input	Feed Input Blob into Network	Detected Object, Category, Confidence

A separate super-category will be maintained alongside additional tips - for example, to notify the user that it is recommended to peel off the bottle label in the event that a plastic bottle is detected. In essence, the model will output the general object category of the detected object, which will then feed into a loaded map that contains special instruction for each super-category.

This data will then be outputted to the user, with the object instance being drawn, alongside confidence level, and the detected category, super-category and special instructions, or advice.

VII. Development environment

A. Platform

Linux

This application will mainly look to support the Linux environment, so that adapting the usage to work on other devices, such as the Raspberry Pi may be possible. However, due to the resource intensive nature of this application, it will mainly focus on supporting the Desktop Linux environment.

B. Programming Language

Python, C++

As the project has a focus on using OpenCV, and the detection and classification of objects, Python will be used in order to process the dataset. C++ may also be leveraged for the OpenCV aspect, but Python may also be used in its stead, as Python has various accessible libraries which helps manipulate images; Inspection, Labeling, and Augmentation. In addition, there are abundant packages to handle and visualize the image.

PostgreSQL

To allow the possibility of expanding and leaving the data accessible, PostgreSQL will be used in order to record the result. In addition, PostgreSQL provides 'Large Object' type that would be more fitting than a normal 'BLOB' type in our application.

C. Development Environment

- Ubuntu 20.04.3
- 2 Cores 4 Threads @ 1.90Ghz
- 12GB DDR4 RAM
- GitHub (Version Control)
- AWS SageMaker (Model Training)

D. Cost Estimation

- AWS SageMaker
- 10 Hours ($0.906 per hour - ml.g4dn.xlarge), 4 Instances
- Estimated (10 * 4) * $0.906 = $36.24

- AWS EFS : $5
- 50GB in total, 20% of which is frequently accessed
- Estimated $4.39 per month

E. Software In Use

SageMaker

SageMaker is an ML integrated platform provided on Amazon Web Services. One of the advantages to use this software is that it provides the accelerated computing resourses. One of the main concerns about our desired model is that there is a large volume of image dataset to train, and the model should be able to provide the various classification results. Due to the compute power of the AWS service, it takes less time to acquire the trained model. Another advantage is that the distributed training is available. Because we can choose how much instances to engage in the training,this feature also reduces the total training time through a parallel processing.

EFS

EFS(Elastic File System) is a cloud storage service of AWS. While the distributed training executes on the cloud, all datasets should be mounted on each instance and be consumed without any concurrent problem, which EFS supports. In addition, its price policy is the number of reading and writing data. As a result, we can efficiently reduce the cost because the total size of required images, including training, validation and augmentation, is quite large, while most of the data remains in "cold storage" during the training.

Mask R-CNN

Mask R-CNN is a DNN that looks to detect segment, or instance objects in an image. Internal components like RPN and FPN can reduce latency. More details below.
Link : https://arxiv.org/abs/1703.06870

Mask R-CNN Tensorflow from AWS projects

Mask R-CNN is based on the Tensorpack project and it's faster than other implementations on the AWS. It's more efficient compared to an equivalent implementation in Keras, in addition to supporting distributed training. More details below.
Link : https://github.com/aws-samples/mask-rcnn-tensorflow

OpenCV

OpenCV is a software library that focuses on computer vision. It contains support for ML model execution, as well as various image manipulation functions. A lot of the UI features will be implemented via OpenCV (ie. bounding boxes, displaying of results, etc), and the model will be fed to OpenCV's native model handling to return these results. It is available on many platforms and languages, supporting Linux, C++ and Python, amongst many others.

VIII. Specifications

A. Input Support

The application will provide an interface in order to connect to a connected camera, to which a connection will then be established. Appropriate feedback will be outputted in the event that the connection is invalid, or fails.
The camera should be able to output a minimum of a 640x480 resolution, and at a stable framerate above 20 FPS. In addition, RGB color division should be supported. Feedback will be handled through the software side, in order to detect when a connection was not able to be established or an unexpected error occurred.
This specification will support the user holding up an object to the camera - with the connection and handling being done via OpenCV. In the event that the input is of a significantly higher resolution, it may be appropriately downscaled and

transformed in order to preserve performance and inference time.

B. Output Feedback

Output feedback will be provided in a couple of different ways. Firstly, a bounding box, or outline will be drawn around each detected object, and appropriately tagged. Secondly, an output will be provided to show what was detected, in addition to its confidence level. Finally, an additional output will be displayed to convey the detected category.
If the application upon startup, or during execution, encounters an exception, the exception will be logged and the user will be notified through the UI. OpenCV will be used in order to display UI elements for the results of the scan, such as the detected category. If the object was unable to be scanned, or scanned with a very low confidence rate, the user will be notified through appropriate GUI elements, also drawn through OpenCV.

C. Realtime Analysis

The model will be trained with 640x640+ resolution COCO annotated input. Based on the detected category, the application will advise on recycling. Classifications, scores and masks for each detection will be outputted per frame. It includes the request and response time with the precision of milliseconds. OpenCV will be used in order to display UI elements such as framerate, classification, score, in addition to drawing the masks/bounding boxes for each detection, alongside inference time.

D. Database and Statistics

The application tracks all detected wastes and collects statistics to a database. Each entry will record the timestamp, in addition to the classification, score/confidence, and detected masks for the predictions. Transactions will be committed upon completion of detection, or upon end of the session.
To provide statistics, the application will collect information regarding the number of the predicted frames, as well as the category, and subcategory that it detected - alongside confidence and other scores, such as the inference time.
Each detection will call an SQL query (PostgreSQL) to insert information resulting from each object scan - the category, subcategory, confidence, inference time, etc. This database may be accessed locally by the user or other applications so that the user may see more statistics on their recycling and scanning habits.

E. Focused Feedback

At a basic level, the application describes the detected object, object category, confidence scoring, and draws a bounding box, or mask around the object.

For detailed matters, the application provides feedback on how to properly dispose of the object in question. For example, it will notify the user if the detected object is not considered to be recycled, based on the Ministry of Environment's posted guidance (eg. If a cigarette is used for the input, it will reply with a notice that cigarettes are not recyclable. If a plastic bottle is held up, it will tip the user that crushing the bottle may be helpful, or to dispose of the label separately).
For this feedback, OpenCV will be used in order to overlay above feedback (ie. drawing a background, and printing text above). The data for recyclability instructions, or tips will be contained in a JSON file, which will be stored in a map/dictionary and referenced upon successfull object inference.

Link: https://me.go.kr/upload/2/editor/202009/18/20200918181806.png
Link: https://me.go.kr/upload/2/editor/202009/18/20200918181814.png