Image-Based Waste Sorting

Anirban Banerjee (aba177), Ayush Raina (ara95), Tanmay Jain (tja34), Siddhartha Haldar (sha285)

Motivation and Background

Recycling is critical for our society and environment - it lowers our carbon footprint, decreases the need for harvesting raw materials, saves energy, reduces greenhouse gases, prevents pollution, and more. British Columbia has one of the most robust recycling systems in North America. Recycle BC collects recyclable waste from over 1.8 million households through curbside, multi-family and depot collection. This covers 155 communities and over 98% of British Columbia's population. Less than one percent of waste collected for recycling is sent overseas. The rest is dealt with locally.

One of the most important steps in recycling is waste segregation, that splits general waste products into different categories to be disposed of, recycled, or composted. However, the enormous volume of waste generated at the community level simply cannot be sorted manually. Automated waste collection systems have long since become a necessity in any endeavor that involves waste disposal. It is estimated that the market for automated waste collection systems will increase at an annual growth rate of 5.9% to reach US$ 365.37 million from 2020 to 2028.

However, there is a second aspect to waste segregation - having waste be sorted by individuals and households *before* being collected. Through programs like Recycle BC, different categories of recyclable waste such as plastic and paper can be sorted into respective bins to be collected every week. This reduces the burden of effort required to sort waste at disposal facilities after collection, and ensures that a larger amount of recyclable material is, in fact, actually recycled. However, it can be quite difficult for the average individual to fully know what category of waste a given item specifically falls under. It may not be immediately obvious how objects like band-aids, syringes, potted plants and jute bags are categorized in terms of waste disposal.

Our motivation in this project is to tackle both problems at once: we aim to create a data solution that takes images of waste items, uses neural networks to label them into one of five waste categories, and outputs a recommendation of how to properly dispose of each waste item. Firstly, our solution can be used at automated waste collection systems to split collected waste on a large scale. For example, waste items on a conveyor belt can be photographed, and our product used to categorize the item. The waste item can then mechanically be diverted into the appropriate line for being disposed of/recycled/composted. Secondly, our solution can be used by individuals to find out how to dispose of their household waste. A user could simply use their smartphone to click a photo of the waste item in question and use our solution - delivered via a web app on their smartphone - to easily find out how to dispose of it.

Problem Statement

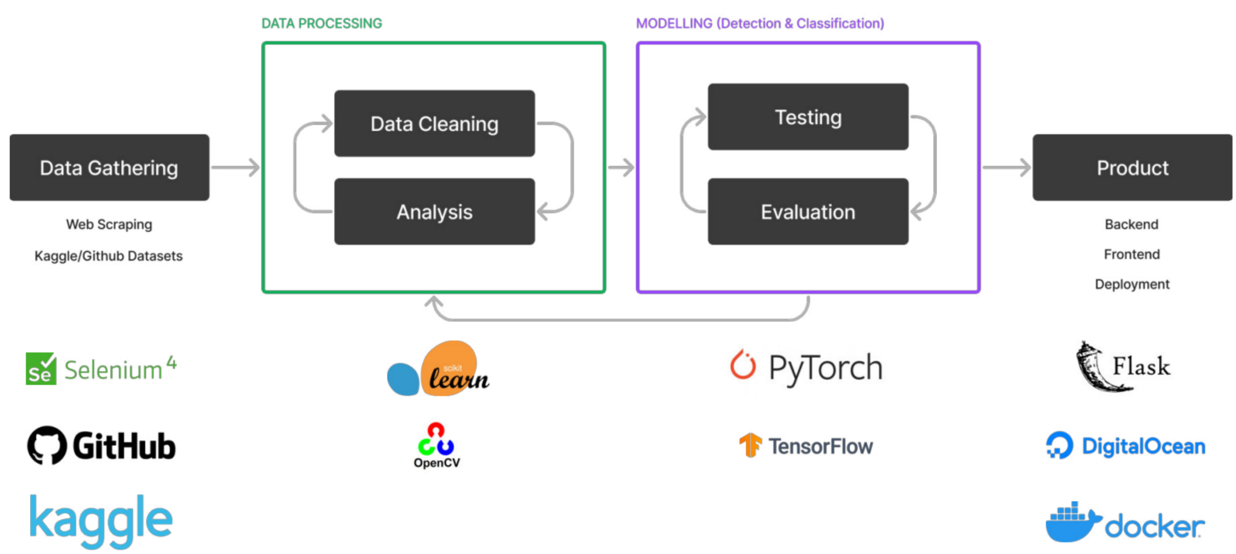
We intended to answer the three following questions through our project:

1. Can we improve waste sorting by using neural networks to identify which waste products are compostable and/or recyclable?
2. Via comparative analysis, which neural network provides the best results for identifying waste? Are some better than others in certain scenarios?
3. Can we improve this process by utilizing active learning procedures?

As we worked on the project, we encountered the following challenges:

1. There is no readily available dataset that contains all the different waste classes and items we wanted to include in our solution. Creating a comprehensive, well-rounded dataset was our first great challenge during this project. This has been explored in more detail in the sections below.
2. Obtaining a dataset with annotations. Initially, our plan was to combine a detection model (that could detect individual items in a photograph containing multiple waste items) with a classification model (that would classify a photograph of a single waste item). However, our detection model relied on using COCO annotations, and of all the datasets we looked through for this project, only the TACO dataset included annotations.
3. Image datasets tend to be large and cumbersome compared to text-only datasets, and training multiple models on our large dataset was a considerable time sink. The slow nature of the training process also meant we couldn't 'course correct' in real time - we would often have to wait long periods of time for the training to complete before choosing our next steps based on the output.
4. When working with some state of the art technologies such as TensorRT, we discovered that due to their newness, there was often relatively little documentation to work with. We had to revise our initial list of detection/classification models based on the challenges we encountered while using some of them.

Data Science Pipeline and Methodology



Data Gathering

We created our own dataset by combining waste image data from a variety of different sources. These primarily consisted of:

1. Annotated images from the TACO dataset, which can be found at http://tacodataset.org/,
2. Multiple non-annotated image datasets from Kaggle and Github listed below,
    a. https://github.com/garythung/trashnet
    b. https://github.com/nikhilvenkatkumsetty/TrashBox
    c. https://www.kaggle.com/techsash/waste-classification-data
    d. https://www.kaggle.com/wangziang/waste-pictures
3. Scraping over 20k pictures from Google Images using an image scraper script.

Through our efforts, we were able to amass a 4GB dataset containing 44,200 items spread out across the 22 different categories listed below:

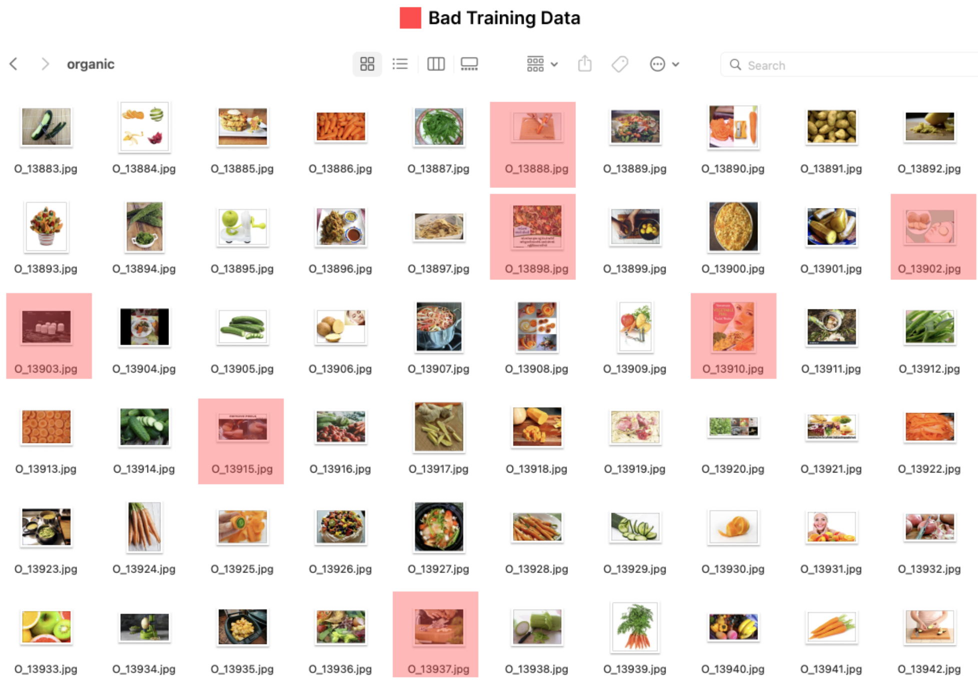
- bandaids
- batteries
- bottles
- bulbs
- cans
- cardboard-boxes
- cigarette-butts
- computers
- cups
- diapers
- electrical-cables
- face-masks
- gloves
- medicines
- organic
- paper
- plastic-bags
- smartphones
- syringes
- tetra-packs
- thermometers
- toothbrushes

Data Cleaning and Analysis

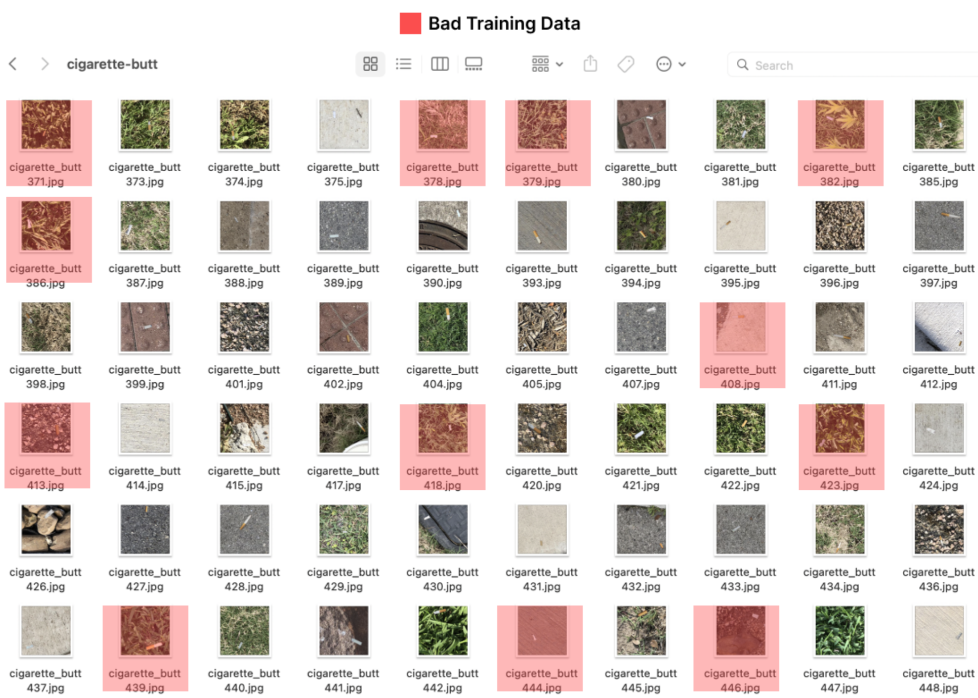
When we first began training our models on our dataset during preliminary testing, we found that they would all produce surprisingly low accuracies. After some investigation, we went through our image data and were surprised to discover that there were a large number of faulty images present. These included cases such as, but not limited to, the ones listed below.

1. Images entirely unrelated to the waste category, such as photographs of moving trucks in the ‘cardboard-boxes’ image set,
2. Images that contained elements unrelated to the waste category, such as photographs of gloved hands holding objects in the ‘gloves’ image set,
3. Images with too many objects in them, such as kitchen countertops in the ‘organic’ image set,
4. Images with text over them, ranging in purpose from watermarks to recipes to memes,
5. Comics, cartoons, drawings, and other artwork.

Example 1 of bad image data:



Example 2 of bad image data:



We also conducted analysis on our images on a per-category basis to ensure that (1) there were enough images present in each category for meaningly model training, and (2) the images were all of usable size. Tiny images might have resulted in poor training, and large images would have caused training to take longer due to their file size. Our observations can be seen in the graph below:

Number of images per waste category:

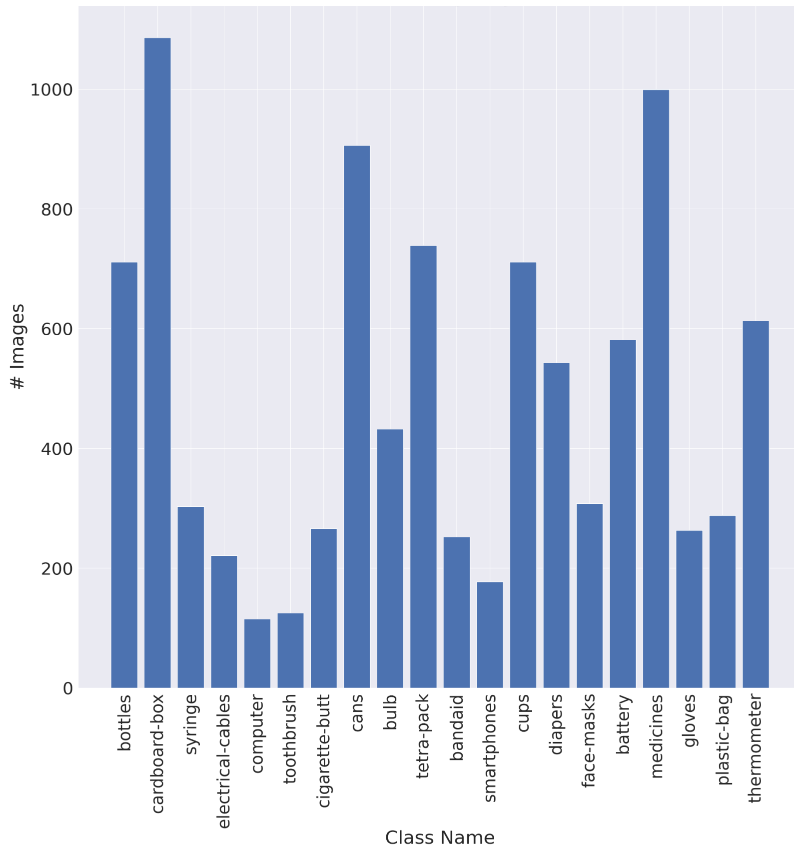
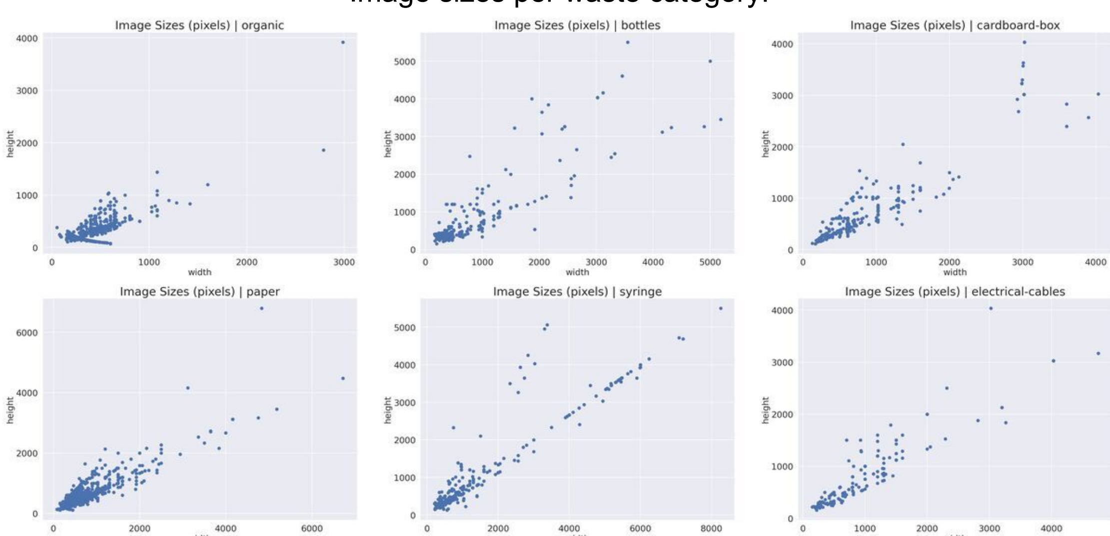


Image sizes per waste category:



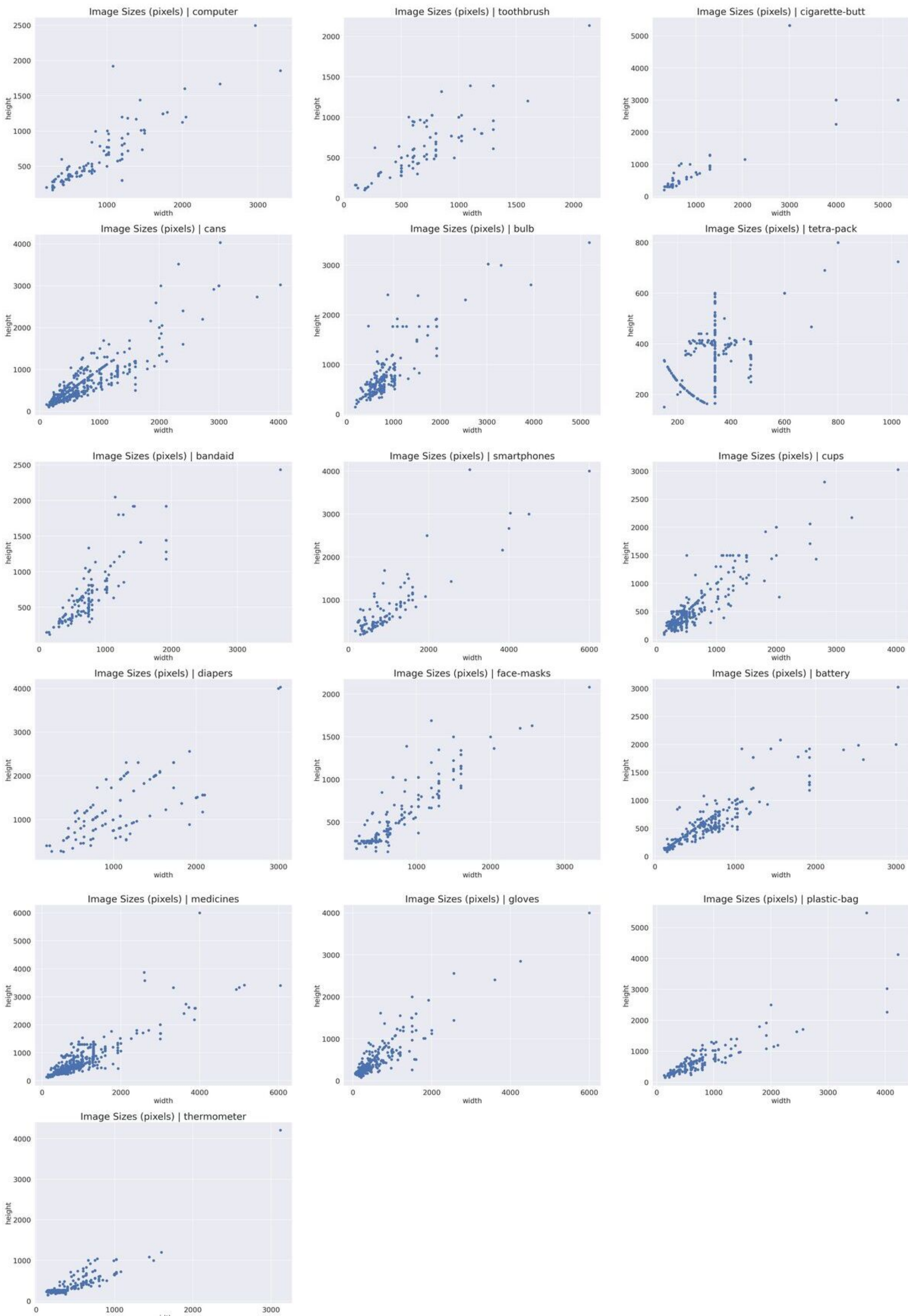


Image Sizes (pixels) | computer
Image Sizes (pixels) | toothbrush
Image Sizes (pixels) | cigarette-butt
Image Sizes (pixels) | cans
Image Sizes (pixels) | bulb
Image Sizes (pixels) | tetra-pack
Image Sizes (pixels) | bandaid
Image Sizes (pixels) | smartphones
Image Sizes (pixels) | cups
Image Sizes (pixels) | diapers
Image Sizes (pixels) | face-masks
Image Sizes (pixels) | battery
Image Sizes (pixels) | medicines
Image Sizes (pixels) | gloves
Image Sizes (pixels) | plastic-bag
Image Sizes (pixels) | thermometer
width
height

After some experimentation on multiple models, we decided to use TensorFlow Pytorch's TorchVision module in order to create a transformation pipeline to standardize images to maximize training effectiveness. In order, we would (1) resize images, (2) convert them to grayscale, (3) transform them into a tensor, and then (4) normalize the tensor. Our code for this transformation pipeline can be seen below:

```
transform = transforms.Compose([
        transforms.Resize((256,256)),
        transforms.Grayscale(num_output_channels=3),
        transforms.ToTensor(),
        transforms.Normalize([0.485, 0.456, 0.406], [0.229, 0.224, 0.225])
    ])
```

Model Testing and Evaluation

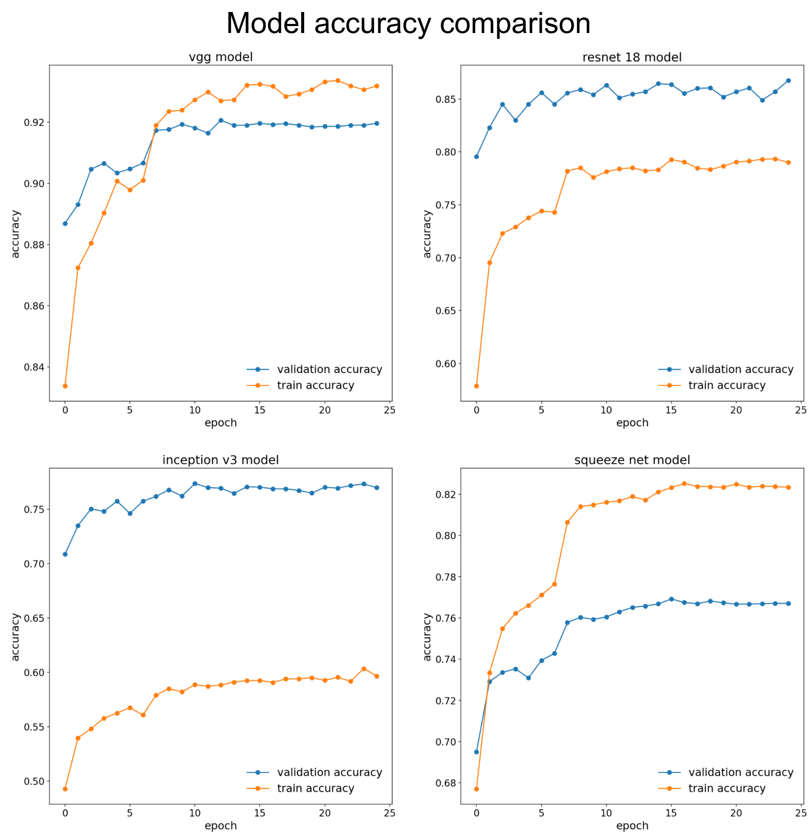
We decided to incorporate both detection and classification models for our waste classifier. The detection model would be used for images containing multiple waste items. The classification model would be used for images containing a single waste item.

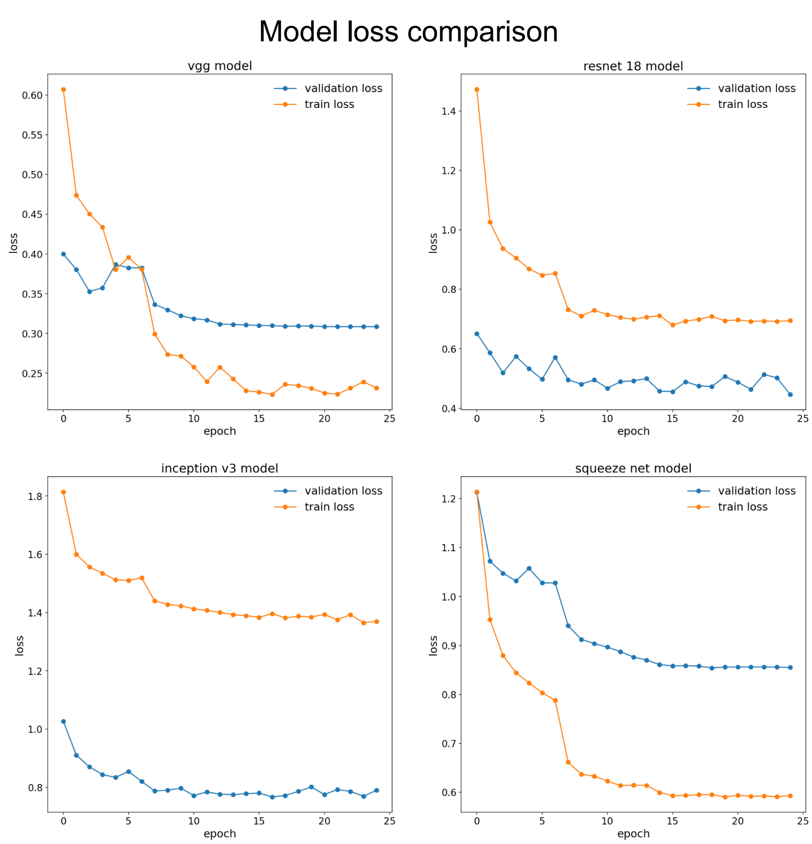
Our envisioned pipeline for waste image analysis consisted of the following steps: first, the user would take a photograph of all their waste items, and the detection model would be run to identify each waste image in the photograph. In the case where the detection model would fail to identify a specific item in the image, the user would be directed to take photographs of those items individually. The classification model would then be run on the individual items to identify them.

We chose the following models for our project. Four different classification models were picked in order to conduct a comparative analysis on their results.

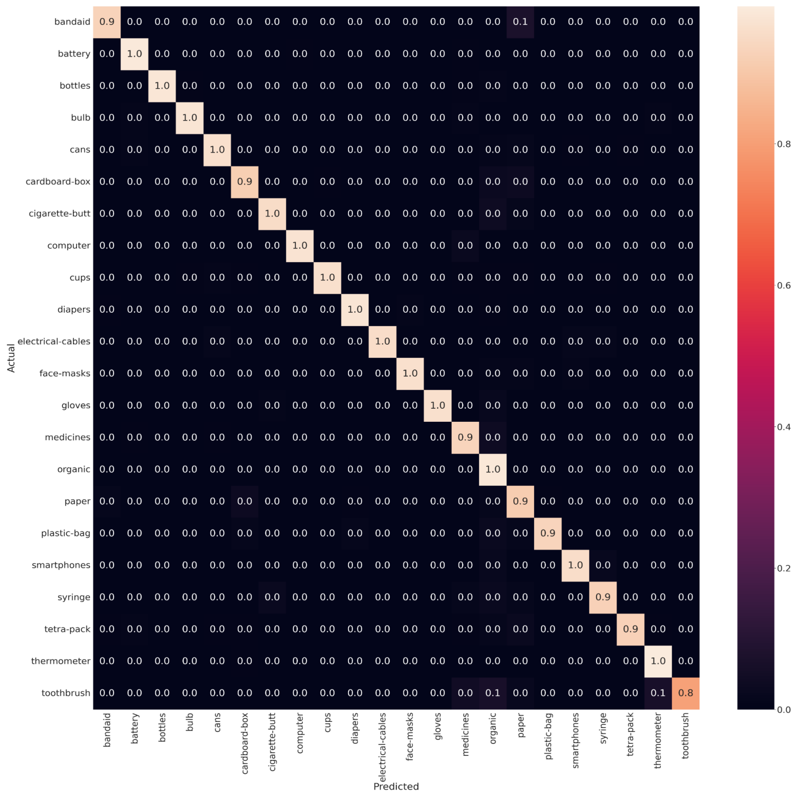
- Detection
  - Selected Mask R-CNN with Resnet50
- Classification
  - Vgg16
  - Resnet18
  - Inception-v3
  - Squeezenet

A comparison of the relative performance of the four classification models has been provided in the graphs below. This comparison includes the model accuracy, model loss, the confusion matrices for each model, and a class-wise accuracy comparison. In general, we see that the Vgg16 model performs the best, with an overall accuracy rate of 93%. We also note that the confusion matrices reveal that all four models decisively identify each waste category and there is virtually zero confusion between classes. This is because our waste items were specifically chosen so as to have different physical shapes and form factors. A more detailed breakdown can be found in the analysis section of our github repository here.

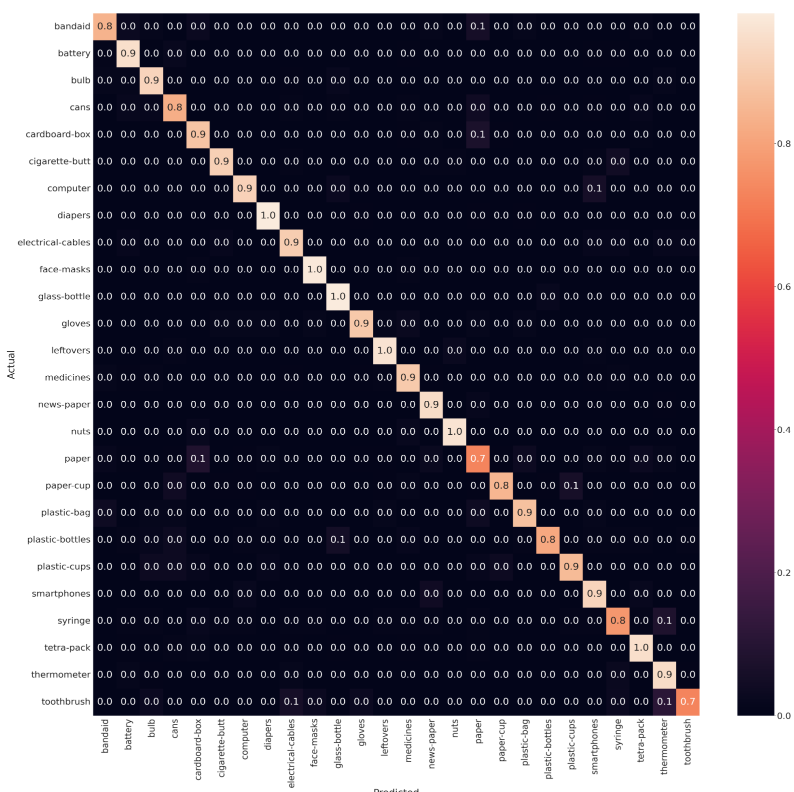

Model accuracy comparison
vgg model
resnet 18 model
inception v3 model
squeeze net model
accuracy
epoch
validation accuracy
train accuracy


Model loss comparison
vgg model
resnet 18 model
inception v3 model
squeeze net model
loss
epoch
validation loss
train loss

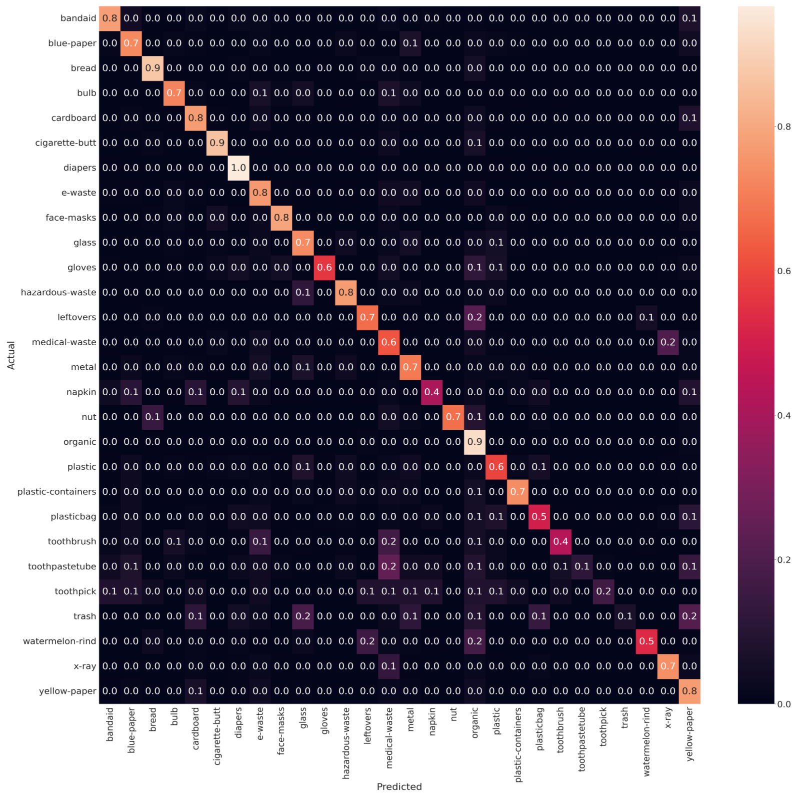
Vgg model confusion matrix



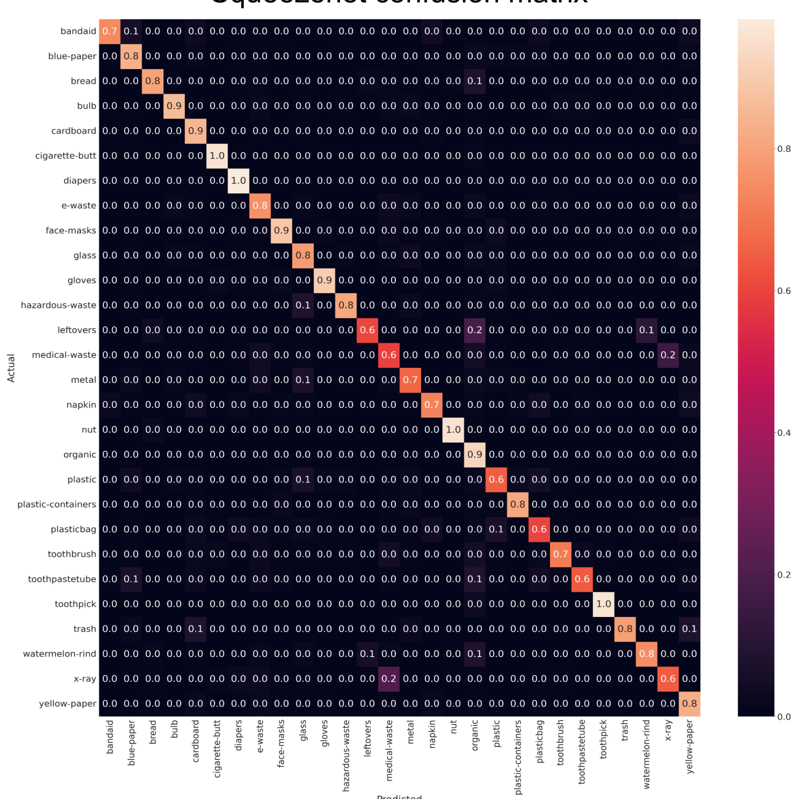
Resnet 18 model confusion matrix



Inception v3 confusion matrix



Squeezenet confusion matrix



Class-wise Accuracy Comparison
bandaid
battery
bottles
bulb
cans
cardboard-box
cigarette-butt
computer
cups
diapers
electrical-cables
face-masks
gloves
medicines
organic
paper
plastic-bag
smartphones
syringe
tetra-pack
thermometer
toothbrush
0.00
0.25
0.50
0.75
1.00
Vgg
Resnet
Inception
Squeeze

Data Product

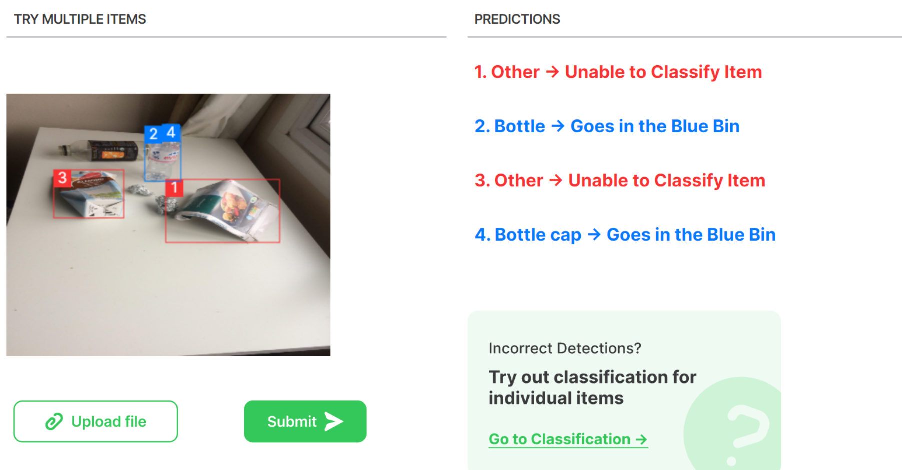
Our data product for this project is a web app for classifying waste images. The app is securely hosted on HTTPS, and is responsive - meaning a user accessing our app on their smartphone can use its camera to directly click photographs of the waste products they want identified. It can be found at the links below:

- Waste Image Detection - to detect and classify multiple waste items in an image
- Waste Image Classification - to classify a single waste item in an image

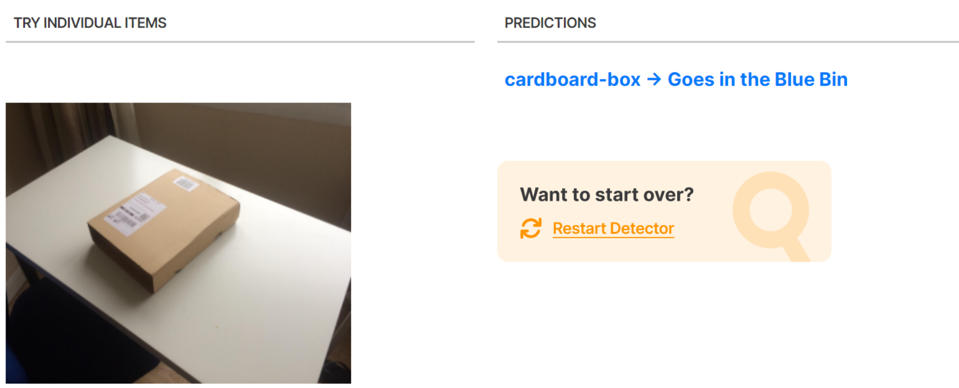
Unfortunately, as a result of us having limited annotated image data to train our model on, waste detection does not succeed as often as waste classification. However, we intend to keep track of every image sent to the web app and store their predictions, creating a data lake on S3. We plan to use this created dataset to apply active learning and retrain our models, making our web app's predictions better over time.

Example of waste detection

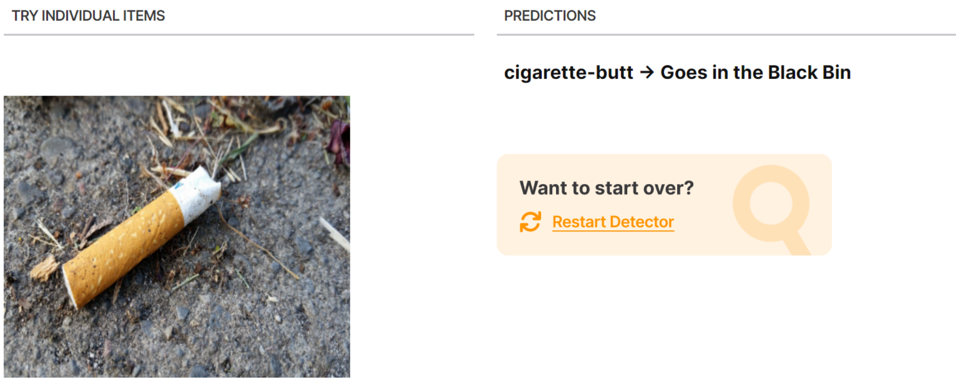
(In a real-world instance, the user would follow the 'go to classification' link and take new photographs of the unclassified items in order to receive disposal advice for them)



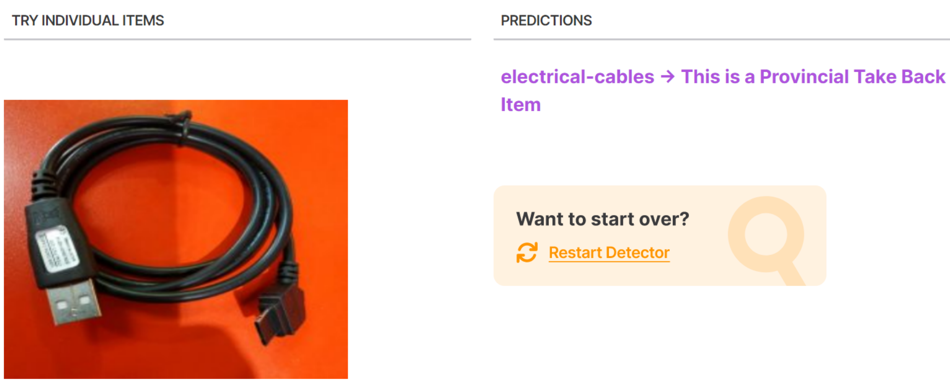
Example of waste classification - blue bin item



Example of waste classification - black bin item



Example of waste classification - provincial take-back item



Lessons Learnt

1. Even when working with pre-existing datasets, outliers and faulty data can still be present. Manual verification and sanity checks should always be performed on datasets to ensure the validity and correctness of the data before using it for any form of analysis or learning.
2. Our experience with faulty data as described above led us to a newfound appreciation of the old adage that garbage in equals garbage out. During preliminary tests, our models would often be stuck at low accuracy values. It was only after manually combing through our data, and finding out about the sheer quantity of poor-quality images that had been included with the pre-existing datasets we had used, that we realized why our models had been performing so poorly. Our effort in painstakingly cleaning the data was rewarded with a higher increase in model accuracy than any other measure we had tried during the entirety of the project.
3. Checkpointing is of critical value when models have long training times. Training our models on the image data we had collected would often take hours to complete, meaning that any interruption or fault would result in large setbacks. Early on, we got into the habit of rigorously checkpointing our progress while training the models in order to avoid losses in case of failure.
4. We gained considerable first-hand technical experience with machine learning deployments and containerization. This project was a useful opportunity to work with multiple classic neural network models, and learn about their advantages and disadvantages in real-world scenarios, including factors such as size, load times, accuracy trade-offs, and more.

5. While developing our web app, we learnt about the invisible value provided by giving our data product a simple, intuitive and visually appealing design with clear and explicit instructions for first-time users. We leveraged our previous experience in the workforce to apply good UI/UX principles in order to create a web app that was not just functional, but also easy and hassle-free to use.

Summary

Our goal in this project was to create a data solution in which neural networks would use images of waste products to identify and sort them into different categories for disposal, recycling, or composting. We also wanted to perform comparative analysis on different pre-trained neural networks to compare and contrast their performance, and see if some were better than others in specific scenarios. Our initial plan was to perform waste image analysis using a combination of detection/classification models: first, we would run detection models on photographs with multiple items, and detect and identify each individual waste item within the picture. If detection failed, we would then run classification models on photographs containing a single item.

We created a 4GB dataset of waste images covering 22 different categories, such as cans, bottles, plastics, cardboard, organic material, electronics, medical waste, etc. We put together this dataset by combining a COCO annotated dataset from TACO, non-annotated datasets from Kaggle/Github/etc, and scraping data for the remaining categories from Google images. After running some initial tests and analysis on the dataset, we found that the data from the pre-existing datasets had a considerable amount of faulty images mixed in with the correct images. To improve the results obtained from training models on our dataset, we manually went through the images and performed data cleaning by identifying and removing faulty images.

We picked Mask R-CNN with Resnet50 as our detection model, and chose four different classification models - Vgg16, Resnet18, Inception-v3 and Squeezenet - to perform comparative analysis on. After training and evaluating the models on our cleaned dataset, our evaluation of model loss and accuracy revealed that the Vgg16 model displayed the best performance, with a 93% accuracy rate. While our image classification pipeline worked extremely well, our image detection pipeline did not perform on par. This was a result of us having limited annotated data to train the detection model with.

Using our trained models, we created a web app, hosted at https://the-bin-conundrum.live/, with a separate classification endpoint at https://the-bin-conundrum.live/classify. We keep track of images sent on the web app and store their predictions, creating a new dataset which we can use to apply active learning and retrain the models and improve their performance over time.

In conclusion, this project was highly beneficial in illustrating the benefits and challenges associated with applying machine learning solutions to a real-world challenge. In addition to creating an in-demand data product with real-world applications, we were able to build a unified and comprehensive dataset of waste images that cover a wide range of waste classes and items. This dataset will also be of invaluable use to future data scientists who wish to explore this topic further.