Computer Vision Project: Smart Waste Model

Team: D103 Group 14

Kate Luong #301464054

Frentzen Taslim #30144920

Instructor: Dr. O.Nilay Yalcin

TA: Minoo Shayaniasab

October 31, 2024

Introduction

In our computer vision project, we aim to develop a supervised classification model to automate waste sorting, specifically distinguishing food organics, paper, plastic, and glass. The problem we are addressing is the inefficiency of traditional waste separation methods, which rely mostly on manual sorting. These approaches are prone to human error, time-consuming, and resource-intensive, leading to improper sorting that disrupts recycling efforts and contributes to environmental pollution. By implementing an automated classification model, we aim to improve recycling accuracy and efficiency.

We believe this model is highly suitable for integration into mobile applications that help users sort waste accurately. For instance, a waste management app could utilize the phone's built-in camera, allowing users to easily take a picture of a waste item. The app would then classify the item as food organics, paper, plastic, or glass, providing instant guidance on proper disposal. This model primarily targets environmentally conscious individuals, families, schools, and organizations dedicated to promoting sustainable practices. Use cases range from personal recycling at home to applications in offices and educational programs. By offering a quick and convenient way to make informed disposal decisions, this model serves as an accessible and practical tool for daily use.

Dataset Report

Image Collection

To create the dataset for the model, we gathered images from various online sources, including free image repositories and public domain datasets like Kaggle. These images represent a diverse range of food organics, paper, plastic, and glass materials, with a total of 2312 images (Google Drive Link: https://drive.google.com/drive/folders/1ABi3Dt42ox_uN-5t34uhogKjNTWn5giD?usp=drive_link).

We sourced a public image classification dataset from Kaggle, titled "RealWaste Image Classification," which contains images of various waste items from an authentic landfill environment. The dataset includes labels that indicate the material type for each image, formatted as "[material_type]_[index].jpg." We downloaded a total of 2,252 images from this dataset, as shown in Figure 1. This includes:

- 921 Files of Plastic
- 500 Files of Paper
- 420 Files of Glass
- 411 Files of Food Organics

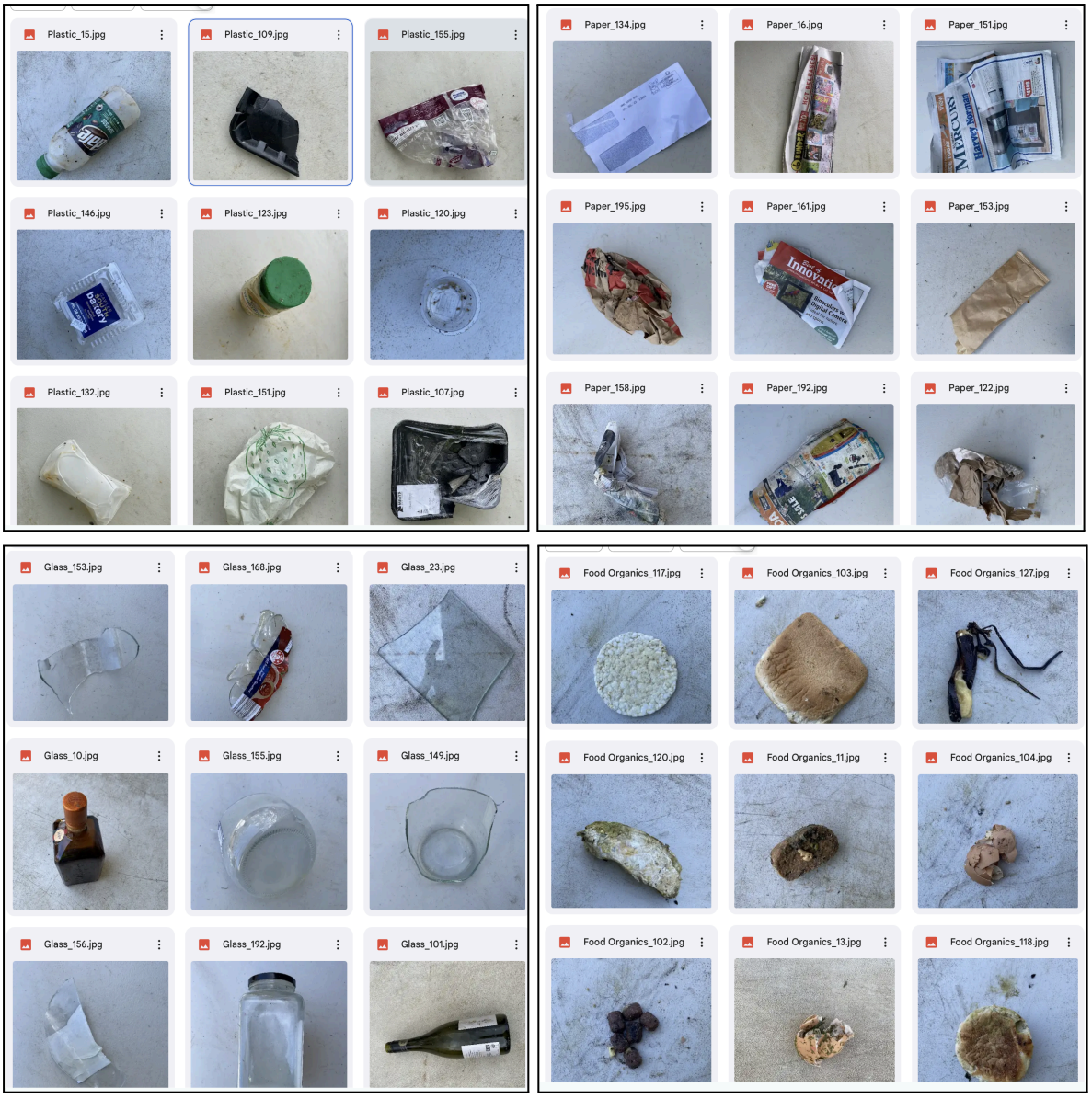

Figure 1. Sample of the 2,252 images retrieved from the "RealWaste Image Classification" dataset on Kaggle, showcasing waste items from a landfill environment, with labels indicating material types (e.g., glass, plastic, food organics, paper). Source: https://www.kaggle.com/datasets/joebeachcapital/realwaste

For the images we gathered independently, we decided to maintain the same labeling format as the online dataset found on Kaggle. We added 60 new images to the dataset (30 images each), evenly distributed across the four material categories, as shown in Figure 2. As a result, 15 images were added to each category, with labels following the same format as the original dataset. This brings the total count of images in the final dataset to:

- 936 Files of Plastic
- 515 Files of Paper
- 435 Files of Glass
- 426 Files of Food Organics

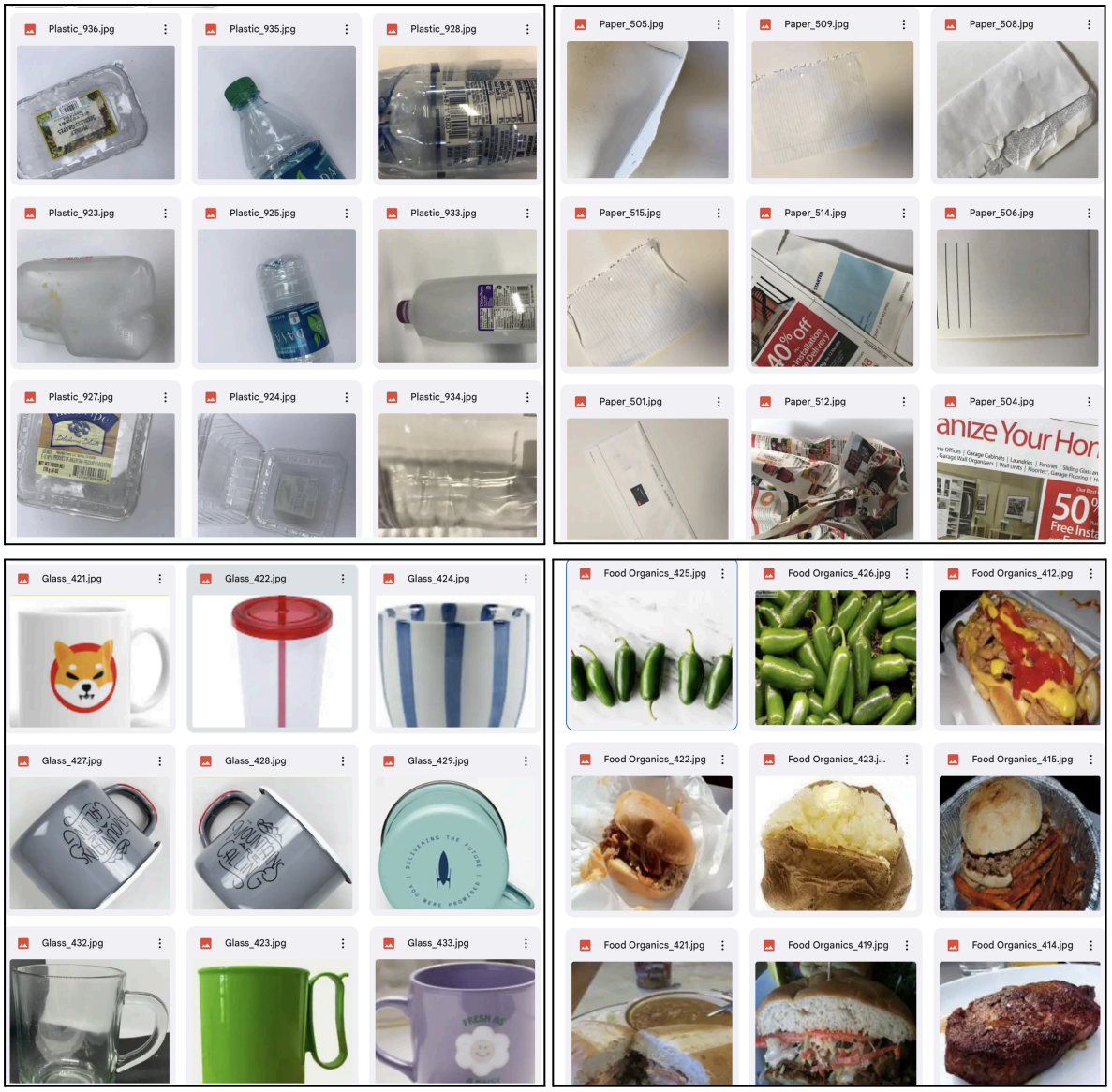

Figure 2. Distribution of the 60 new data gathered for the four material categories, with 15 images added to each category (glass, plastics, food organics, and paper).

Organization & Labeling

To organize the dataset for training and testing, we created a folder structure within Google Drive, which was then mounted in Colab. The overall dataset was divided into two main directories: /dataset/train and /dataset/test. Under each of these directories, we created subfolders for each material type: /Plastic, /Paper, /Food Organics, and /Glass.

Figure 3 and shows how the images are sorted by category, with each folder containing images corresponding to a specific material type as seen in Figure 3 and 4. This folder-based structure serves as the labeling system for the classification

task, allowing the model to distinguish between the different categories during training and testing.

```
train_dir = '/content/drive/MyDrive/CV-Project/train',
test_dir = '/content/drive/MyDrive/CV-Project/test',

# Create the main dataset directories
os.makedirs('/content/drive/MyDrive/CV-Project/dataset/train', exist_ok=True)
os.makedirs('/content/drive/MyDrive/CV-Project/dataset/test', exist_ok=True)

# Create subdirectories for train dataset
os.makedirs('/content/drive/MyDrive/CV-Project/dataset/train/Plastic', exist_ok=True)
os.makedirs('/content/drive/MyDrive/CV-Project/dataset/train/Paper', exist_ok=True)
os.makedirs('/content/drive/MyDrive/CV-Project/dataset/train/Food Organics', exist_ok=True)
os.makedirs('/content/drive/MyDrive/CV-Project/dataset/train/Glass', exist_ok=True)
```

Figure 3. Code to create the folder structure for 'Training' and 'Testing' dataset with subdirectories for each material type.

```
#create path to existing data folder
plastic_folder = '/content/drive/MyDrive/CV-Project/RealWaste/Plastic'
paper_folder = '/content/drive/MyDrive/CV-Project/RealWaste/Paper'
organics_folder = '/content/drive/MyDrive/CV-Project/RealWaste/Food Organics'
glass_folder = '/content/drive/MyDrive/CV-Project/RealWaste/Glass'

#creating path to our training folder
plastic_train_folder = '/content/drive/MyDrive/CV-Project/dataset/train/Plastic'
paper_train_folder = '/content/drive/MyDrive/CV-Project/dataset/train/Paper'
organics_train_folder = '/content/drive/MyDrive/CV-Project/dataset/train/Food Organics'
glass_train_folder = '/content/drive/MyDrive/CV-Project/dataset/train/Glass'

#creating path to our testing folder
plastic_test_folder = '/content/drive/MyDrive/CV-Project/dataset/test/Plastic'
paper_test_folder = '/content/drive/MyDrive/CV-Project/dataset/test/Paper'
organics_test_folder = '/content/drive/MyDrive/CV-Project/dataset/test/Food Organics'
glass_test_folder ='/content/drive/MyDrive/CV-Project/dataset/test/Glass'
```

Figure 4. Code for defining paths to source and destination folders for 'Training' and 'Testing' data.

We created a function that organizes images from a source directory into training and testing folders as seen in Figure 5. It first retrieves all image filenames from the source directory, and then splits them into 80% of the images allocated for training and 20% for testing. It then iterates through both the training and testing image lists, checking if the files are in `.jpg` format. For each image, the code opens the image, converts it to RGB mode, and saves it in the respective training or testing folder. The function is called for each category of images (Plastic, Paper, Food Organics, Glass), organizing them into their respective train and test directories.

```python
def organize_images(source_folder, train_folder, test_folder):

    # Get all files from the source directory
    all_images = os.listdir(source_folder)
    train_images, test_images = train_test_split(all_images, test_size=0.2, random_state=50)

    for image_file in train_images:
        if image_file.endswith(".jpg"):
            image_path = os.path.join(source_folder, image_file)
            image = Image.open(image_path)
            image = image.convert("RGB")
            image.save(os.path.join(train_folder, image_file), "JPEG")

    for image_file in test_images:
        if image_file.endswith(".jpg"):
            image_path = os.path.join(source_folder, image_file)
            image = Image.open(image_path)
            image = image.convert("RGB")
            image.save(os.path.join(test_folder, image_file), "JPEG")

# Organize each dataset
organize_images(plastic_folder, plastic_train_folder, plastic_test_folder)
organize_images(paper_folder, paper_train_folder, paper_test_folder)
organize_images(organics_folder, organics_train_folder, organics_test_folder)
organize_images(glass_folder, glass_train_folder, glass_test_folder)
```

Figure 5. Function to organize and split images from the source directory into training and testing sets and saved into the appropriate subfolders.

Training Report

YOLOv8 Model Training

We train the YOLOv8 classification models with different epochs to observe how training duration affects model performance. To start off, we used a pre-trained YOLOv8 classification model (yolo8n-cls.pt) and set different epochs (5, 10, and 20) for training. Each model was trained using the same dataset but varied in training intensity to observe improvements and avoid underfitting or overfitting.

To validate the model to assess metrics for accuracy and speed, after the model was trained, we validated it using the test dataset, capturing the Top-1 and Top-5 accuracies and the model's processing speed. These metrics provide insight into model's overall reliability and efficiency.

After that, we evaluate the model predictions and visualize classification accuracy using a confusion matrix. Each model will generate predictions and those predictions will be compared with the actual test labels to form a confusion matrix, which illustrates where each model is performing well or struggling across the class labels.

Lastly, we test our model performance in the validation using 3 batches of 16 images in each model to see if it can predict the label correctly.

Model Performance

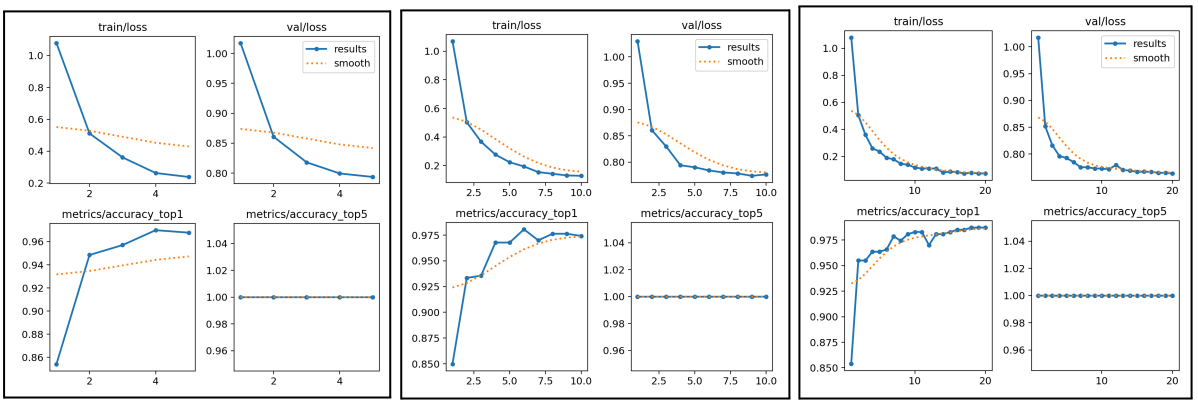

Figure 6. Performance curves comparing the results of Model 1 (trained for 5 epochs), Model 2 (trained for 10 epochs), and Model 3 (trained for 20 epochs), presented from left to right.

epoch	time	train/loss	metrics/accuracy_top1	metrics/accuracy_top5	val/loss	lr/pg0	lr/pg1	lr/pg2
1	176.412	1.0794	0.85408	1	1.01747	0.000235948	0.000235948	0.000235948
2	333.703	0.51157	0.9485	1	0.8608	0.000380107	0.000380107	0.000380107
3	486.295	0.36101	0.95708	1	0.81809	0.000430017	0.000430017	0.000430017
4	639.551	0.26315	0.96996	1	0.79946	0.000289884	0.000289884	0.000289884
5	792.094	0.23818	0.96781	1	0.79361	0.000148512	0.000148512	0.000148512

Figure 7. Performance metrics for Model 1 (5 epochs) of training and validation loss, Top-1 and Top-5 accuracy, and learning rates for different parameter groups.

epoch	time	train/loss	metrics/accuracy_top1	metrics/accuracy_top5	val/loss	lr/pg0	lr/pg1	lr/pg2
1	154.632	1.07074	0.84979	1	1.02994	0.000235948	0.000235948	0.000235948
2	303.233	0.50311	0.93348	1	0.8608	0.000427027	0.000427027	0.000427027
3	449.436	0.36662	0.93562	1	0.82988	0.000570983	0.000570983	0.000570983
4	598.903	0.27657	0.96781	1	0.79408	0.000501942	0.000501942	0.000501942
5	744.65	0.22279	0.96781	1	0.79029	0.000431256	0.000431256	0.000431256
6	890.637	0.19354	0.98069	1	0.78428	0.00036057	0.00036057	0.00036057
7	1038.58	0.15378	0.96996	1	0.78041	0.000289884	0.000289884	0.000289884
8	1185.26	0.14216	0.97639	1	0.77862	0.000219198	0.000219198	0.000219198
9	1333.79	0.13068	0.97639	1	0.77405	0.000148512	0.000148512	0.000148512
10	1480.44	0.12785	0.97425	1	0.77672	7.7826E-05	7.7826E-05	7.7826E-05

Figure 8. Performance metrics for Model 2 (10 epochs) of training and validation loss, Top-1 and Top-5 accuracy, and learning rates for different parameter groups.

epoch	time	train/loss	metrics/accuracy_top1	metrics/accuracy_top5	val/loss	lr/pg0	lr/pg1	lr/pg2
1	149.808	1.0794	0.85408	1	1.01747	0.000235948	0.000235948	0.000235948
2	298.384	0.50546	0.95494	1	0.85157	0.000450488	0.000450488	0.000450488
3	444.695	0.36088	0.95494	1	0.81624	0.000641465	0.000641465	0.000641465
4	589.147	0.26054	0.96352	1	0.79622	0.000607971	0.000607971	0.000607971
5	736.235	0.2366	0.96352	1	0.79265	0.000572628	0.000572628	0.000572628
6	882.28	0.19015	0.96567	1	0.78493	0.000537285	0.000537285	0.000537285
7	1029.87	0.17971	0.97854	1	0.77533	0.000501942	0.000501942	0.000501942
8	1176.61	0.14605	0.97425	1	0.77556	0.000466599	0.000466599	0.000466599
9	1323.71	0.13947	0.98069	1	0.77291	0.000431256	0.000431256	0.000431256
10	1469.66	0.11847	0.98283	1	0.77299	0.000395913	0.000395913	0.000395913
11	1616.44	0.11114	0.98283	1	0.77161	0.00036057	0.00036057	0.00036057
12	1762.47	0.11152	0.96996	1	0.77952	0.000325227	0.000325227	0.000325227
13	1908.82	0.1112	0.98069	1	0.77012	0.000289884	0.000289884	0.000289884
14	2052.76	0.08177	0.98069	1	0.76917	0.000254541	0.000254541	0.000254541
15	2200.12	0.08801	0.98283	1	0.76654	0.000219198	0.000219198	0.000219198
16	2346.7	0.08406	0.98498	1	0.76712	0.000183855	0.000183855	0.000183855
17	2493.22	0.07487	0.98498	1	0.76682	0.000148512	0.000148512	0.000148512
18	2638.88	0.0801	0.98712	1	0.76492	0.000113169	0.000113169	0.000113169
19	2785.02	0.07413	0.98712	1	0.76492	7.7826E-05	7.7826E-05	7.7826E-05
20	2930.7	0.07553	0.98712	1	0.76361	4.2483E-05	4.2483E-05	4.2483E-05

Figure 9. Performance metrics for Model 3 (20 epochs) of training and validation loss, Top-1 and Top-5 accuracy, and learning rates for different parameter groups.

All models showed a consistent decrease in both training and validation losses as training progressed, indicating that they effectively learned from the training data and generalized well to the validation set.

Training Loss: Model 1 has a training loss of 23.818%, Model 2 reaches 12.785%, and Model 3 achieves the lowest at 7.553%. This shows that Model 3 optimized its parameters the best, making better predictions compared to the other models.

Validation Loss: Model 1 has a validation loss of 79.361%, Model 2 reaches 77.672%, and Model 3 reaches the lowest at 76.361%. Again, this indicates that Model 3 was the most effective in learning complex patterns, leading to a better ability to generalize to unseen data compared to Model 1.

Top-1 Accuracy: This metric measures the model's primary prediction accuracy. As the number of epochs increased, we observed a steady improvement in accuracy in all three models' learning curves in Figure 6.

All models demonstrated a significant rise in Top-1 accuracy, indicating that with more training, the models learned to classify the data more accurately.

According to Figure 7-9, Model 1 achieves a Top-1 accuracy of 96.78%, Model 2 reaches 97.425%, and Model 3 achieves 98.71%. This indicates that Model 3, with more epochs, demonstrates the best ability to generalize by effectively learning and extracting complex relationships within the data.

Top-5 Accuracy: This metric indicates whether the correct class appeared within the top five predictions, which helps us assess the model's ability to generalize, especially when distinguishing between nuanced classes. All models achieved 100% Top-5 accuracy according to Figure 6-9. This means that the correct class was always among the top five predictions, demonstrating that the models were consistently able to make highly accurate predictions even in cases with more complex or subtle class distinctions.

Speed: With more epochs, the processing time slightly increases reflecting a trade-off between speed and accuracy. Longer training times helped reduce validation and training loss, showing all three models were able to learn more complex patterns in the data, which enhanced generalization to unseen data.

Underfitting: Models 2 and 3 did not show signs of underfitting, as indicated by the sharp downward curve in their loss functions over time. This suggests that these models were effectively learning complex patterns and generalizing well to unseen data. However, Model 1 exhibited a flat loss curve at the beginning, indicating difficulty in capturing the underlying patterns in the data and most likely underfitting.

Overfitting: Overfitting occurs when the model memorizes the training data too well and in our case, this was observed with an increase followed by a decrease in Top-1 accuracy. This behavior could be due to the small training dataset, which might have limited the model's ability to generalize efficiently.

Confusion Matrix

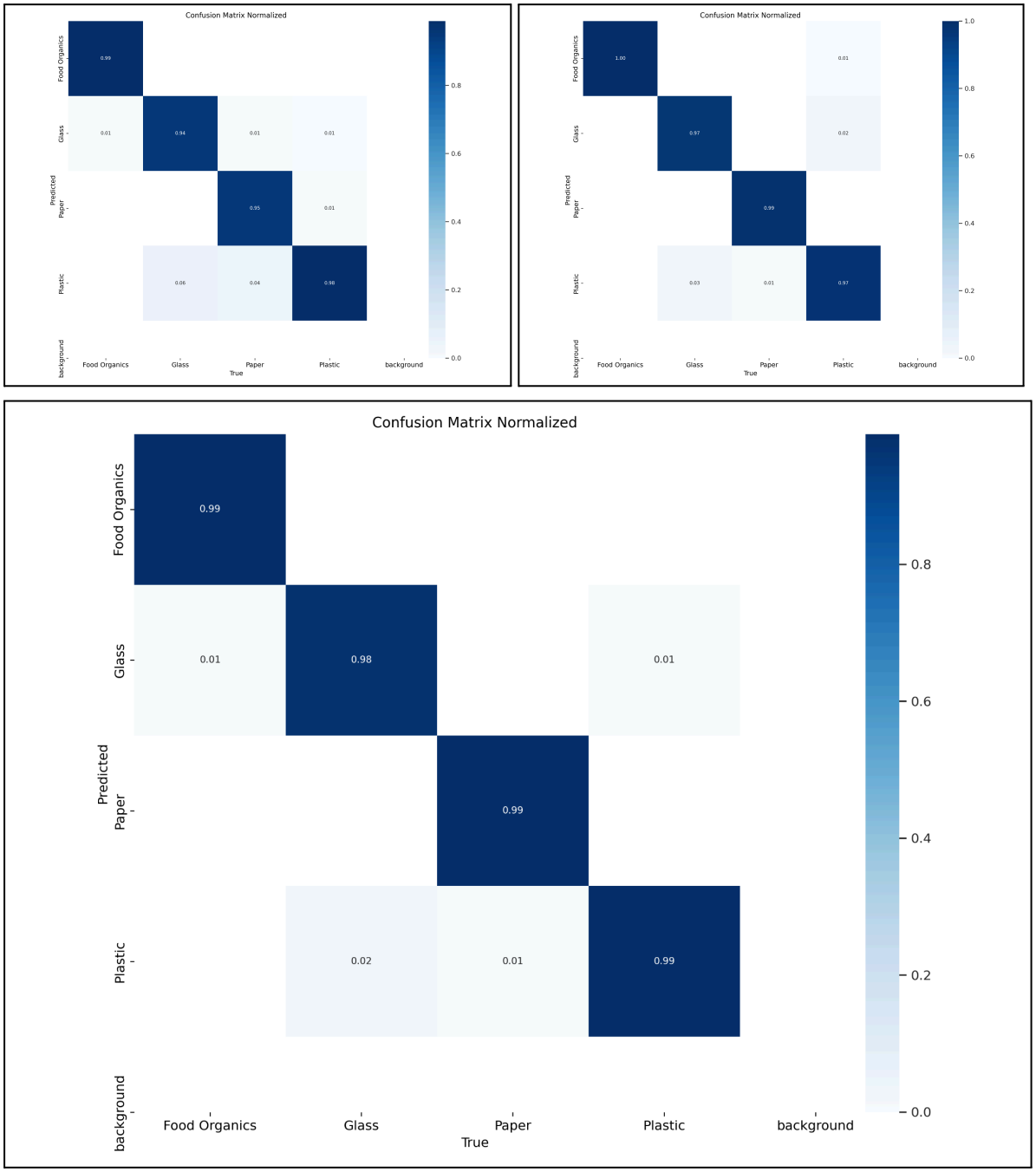

Figure 10. Confusion Matrix comparing the performance of Model 1 (trained for 5 epochs), Model 2 (trained for 10 epochs), and Model 3 (trained for 20 epochs), arranged from left to right.

For all three models, the true positive rate for food organics, paper, and plastic is 99%, indicating that the model accurately classifies nearly all items in these categories. However, as seen in Figure 10, there are a few false negatives in each of these categories:

- 1% of Food organics is often misclassified as Glass in Model 1 and Model 2.
- Glass is often misclassified as Plastic in all three models (6% in Model 1, 3% in Model 2, and 1% in Model 3).
- Paper is also misclassified as Plastic in all three models (4% in Model 1, 1% in both Model 2 and 3).
- Plastic is accurately identified in all three models, but it is most commonly misclassified as Glass, with (1% in Model 1, 2% in Model 2, and 1% in Model 3).

Comparison of the three models: We observed that increasing the number of epochs generally improves the model's true positive rates, as it allows the model to better learn and identify patterns for accurate image classification. However, an interesting trend we observed is that Model 1 achieved a true positive rate of 98%, while Model 2 achieved 97%. This slight decline in performance suggests that as the number of epochs increases, the model may begin to overfit, becoming too specialized in the training data and less capable of generalizing effectively to unseen data.

Although Model 3 does not perfectly classify all categories, it demonstrates a strong ability to classify waste items with minimal errors. The misclassifications seem to be primarily due to shared visual features—some plastic items resemble glass, leading to confusion. In contrast, categories with distinct visual characteristics (e.g., food organics vs. paper or plastic) are less prone to misclassification.

Batch Predictions

Figure 11. Comparison of actual labels (left) and predicted labels (right) for the first batch in Model 1.

Figure 12. Comparison of actual labels (left) and predicted labels (right) for the second batch in Model 1.

Figure 13. Comparison of actual labels (left) and predicted labels (right) for the third batch in Model 1.

Figure 14. Comparison of actual labels (left) and predicted labels (right) for the first batch in Model 2.

Figure 15. Comparison of actual labels (left) and predicted labels (right) for the second batch in Model 2.

Figure 16. Comparison of actual labels (left) and predicted labels (right) for the third batch in Model 2.

Figure 17. Comparison of actual labels (left) and predicted labels (right) for the first batch in Model 3.

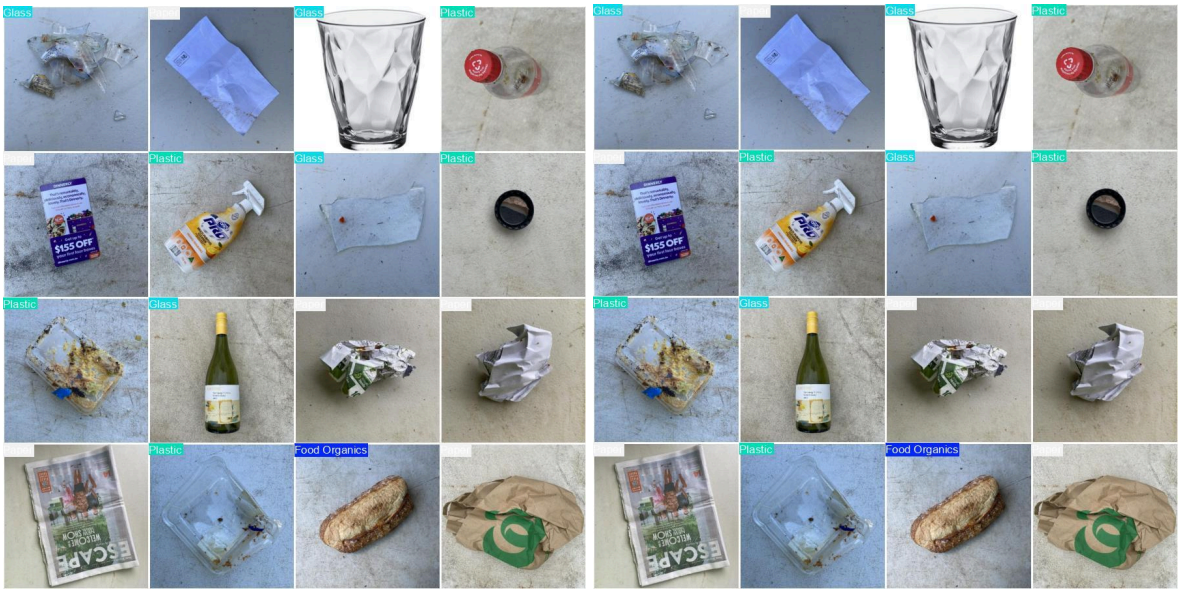

Figure 18. Comparison of actual labels (left) and predicted labels (right) for the second batch in Model 3.

Figure 19. Comparison of actual labels (left) and predicted labels (right) for the third batch in Model 3.

Both model 2 and 3 correctly classify all the waste items in their three batches as see in Figure 14-19. On the other hand, Model 1 correctly identified 15 out of 16 waste items. The only misclassification occurred with the fourteenth item in the third batch of Model 1, which was labeled as glass instead of plastic seen in Figure 13. This misclassification is likely due to the visual similarities between plastic and glass.

Potential Bias Issues

Our dataset might exhibits some bias due to **class imbalance**, with a disproportionate number of images of plastic items compared to other classes. This imbalance could cause the model to favor the plastic category, resulting in more accurate predictions for plastic waste while potentially underperforming on categories with fewer images, such as glass, food organics, and paper.

This is evident in the confusion matrix and the third batch of Model 1, which shows that the model achieves better accuracy with the plastic class, indicating it may be biased toward this material. Consequently, the model's performance is limited in predicting less-represented waste categories accurately, which reduces its generalizability.

The inclusion of images that we individually found also contain some that are sourced from online websites, particularly PNG images with plain backgrounds. This could introduce a form of bias into the model's training and evaluation process. These images likely have a very different visual context compared to the real-world trash images we captured in our own homes as well as in the Kaggle dataset. This may include more complex backgrounds, varying lighting conditions, and diverse textures. This discrepancy between the two types of images could cause the model to learn features that are not fully representative of real-world scenarios. For example, the model may overemphasize the contrast of items against simple backgrounds, making it more likely to misclassify items in cluttered, real-world environments.

To mitigate these biases, firstly we could balance the dataset by augmenting underrepresented classes or collecting additional images to ensure each class has a similar number of samples, allowing for more equal learning across waste types. Secondly, it is crucial to ensure that the training dataset is diverse and representative by including only images from real-world settings with a variety of backgrounds and lighting conditions.

Potential Privacy Issues

There are notable privacy concerns within the dataset. First, **identifiable information** is visible in some of the images, such as personal information on envelopes, which raises privacy risks by potentially exposing sensitive details. Secondly, because some images were taken at our homes, there is an element of **location privacy** risk, as these images might inadvertently reveal aspects of our personal environment and lifestyle. Thirdly, because we didn't do a close examination of the dataset, some captured items in the dataset could **reveal private or sensitive information** about individuals' habits, preferences, or medical

conditions. This could be within discarded medication containers, product packaging, or confidential documents.

To address these issues, a thorough manual inspection of each image would be necessary to identify any inadvertently captured private or sensitive information. For larger datasets, we can automate the process by flagging images with potentially sensitive content, such as medicine containers or documents, for further review or removal. During the manual review, identifiable information should be obscured by blurring or cropping to ensure privacy is maintained.

Our Challenges During the Process

We faced difficulties collaborating in Google Colab, as setting up a Google Drive folder to store our datasets allowed access to files from only one account. Additionally, Colab did not auto-save changes from both users, which led to version conflicts and caused some code updates to be lost. To overcome these challenges, we decided to divide responsibilities by assigning specific tasks to each team member. One member focused on coding the sections of the model, while the other handled the result analysis.

Another issue we encountered was the long waiting time for model training, particularly with Model 3, which required 20 epochs and took over 30 minutes to complete. This prolonged training time caused significant lagging issues on the computer. Despite these delays, the results were successfully saved without any data corruption. To mitigate this in the future, we could consider optimizing the model or using more powerful hardware to speed up the training process and reduce lag.

Conclusion

In conclusion, our models demonstrated significant improvements in classification accuracy as the number of epochs increased, with higher epochs allowing the model to better understand patterns and classify waste items more effectively. Notably, the model's performance shows little progress after 10 epochs, suggesting that further training beyond this point may not yield significant gains in differentiation between categories. However, there were some shortcomings observed, such as misclassifications due to visual similarities between certain categories, like glass and plastic. Some factors could include noisy or complex conditions, such as poor lighting or dirt on items, which limited its ability to predict accurately in real-world settings.

From an ethical perspective, misclassifications can have environmental consequences, especially if materials like glass are mistaken for plastic, potentially leading to improper waste disposal and harming the environment. Additionally,

over-reliance on the model may hinder users' understanding of the characteristics of different waste categories, making it more difficult to detect misclassifications. These ethical concerns highlight the importance of using such models in conjunction with human judgment, rather than as the sole decision-making tool in waste management processes.

Link to our Github:
https://github.com/katelnn/CV-Project-Smart-Waste-Model.git

References

Arvidsson, J. (2024, January). *RealWaste Image Classification*. Kaggle. https://www.kaggle.com/datasets/joebeachcapital/realwaste

glass Computer Vision Project. (n.d.). Roboflow Universe. https://universe.roboflow.com/dogcat-1rqol/glass-rkz6m

trash classification Computer Vision Project. (n.d.). Roboflow Universe. https://universe.roboflow.com/recycle-vmmzf/trash-classification-5nhrf-syjsn