Garbage Classification

Assisting Recycling Plants in Sorting Waste

Team: Garbage Pail Kids
Ayda Nayeb Nazar, Ghiwa Lamah, Kelly Zabors

School of Information at the University of California, Berkeley
Data Science 281 – Computer Vision
Professor Rachel Brown

December 15th, 2023

Abstract

In this project we explore the potential to implement computer vision as a method of automated waste sorting for recycling plants. We use the Garbage Classification dataset sourced from Kaggle, consisting of images of commonly recycled items from 12 object classes. We experimented with various features across the images including color, edges, and HOG, with logistic regression and the pre-trained neural network ResNet50. We end by reducing dimensionality, exploring the trade off between accuracy and efficiency, and finally our recommendation for implementing at scale.

1. INTRODUCTION

"Recycling in the US is broken. How do we fix it?"[1]

Despite the increase in awareness of the importance of recycling, contamination of recycled material remains one of the key challenges of consistent, effective recycling. If the material isn't sorted correctly, a small contamination could prohibit a large batch of material from being recycled. With this challenge in mind, this project aims to explore image classification as a solution to sort different categories of garbage and recyclable materials.

2. DATASET & PREPROCESSING

The Garbage Classification dataset, obtained from Kaggle, contains 15,150 labeled images from twelve classes of garbage: batteries, biological, brown-glass, cardboard, clothes, green-glass, metal, paper, plastic, shoes, trash, and white-glass. Appendix 1 shows sample images from each class. Most classes contained between 600-1000 images on average, with the exception of clothes and shoes. Both classes had significantly more images than the average class, causing an imbalance in the dataset; as can be seen in Figure 1.

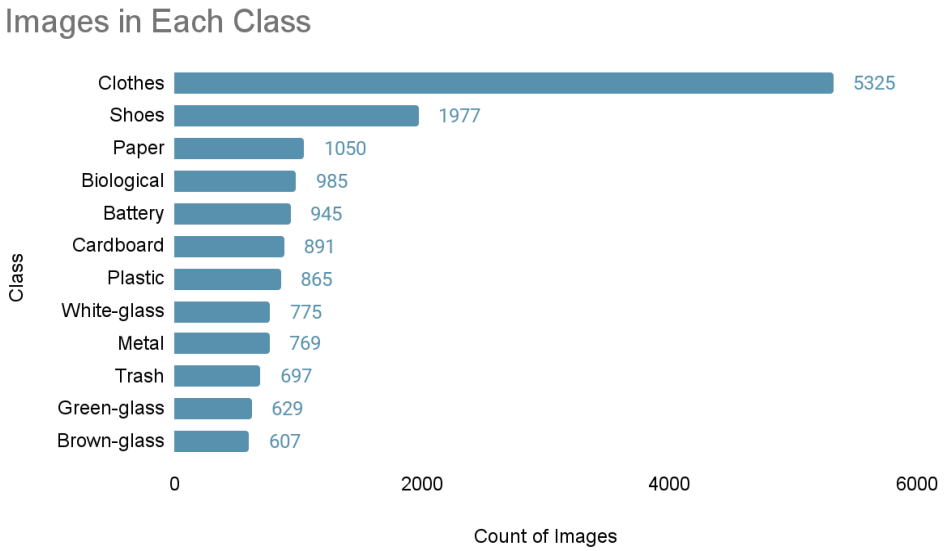


Figure 1: Count of Images in Each Class

[1] A 2020 article by the Columbia Climate School at Columbia University
https://news.climate.columbia.edu/2020/03/13/fix-recycling-america/

While some imbalance in image quantity is expected, we were concerned the severe imbalance in the clothes and shoes categories would lead the models to exhibit bias towards the two majority classes. To manage this, we randomly sampled 1,000 images from each of the two aforementioned classes to ensure a more equal proportion to feed to our classifiers. To resolve variations in dimensions of the images, we resized all images to 200x200.

At this stage, our guess for useful image features starts with edges and corners. Objects with more rigid structure like cardboard should have very distinct edges, compared to objects like clothing or biological waste that should have softer corners as a distinction.

3. FEATURE EXTRACTION

While we explored numerous feature vector options for our dataset, we learned that some features did not particularly work well to reduce the dimensionality of our dataset. The six features extraction techniques we explored were image edge detection, image corner detection, histogram of oriented gradients (HOG), bag of visual words (BOVW), bag of visual colors (BOVC), and Residual Network (ResNet).

For edge, corner, and HOG feature extraction, we converted the images to grayscale to reduce dimensionality and implemented Gaussian blurring for noise reduction. Figure 2 shows this process implemented on a sample of images in our dataset.

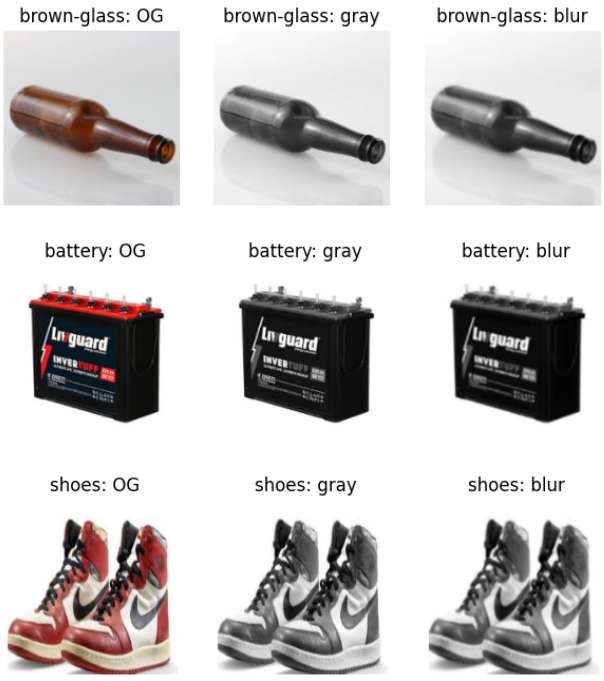


Figure 2: Original Images (OG), grayscale (gray) and Gaussian Blur (blur)

3.1 EDGE AND CORNER DETECTION

Due to the nature of dataset having differentiated objects, some of which maintain relatively consistent shapes and patterns (examples such as bottles, batteries, shoes as shown in Figure 2 above), we expected that edge and corner detection would sufficiently reduce image dimensionality while highlighting key object boundaries to help identify different object classes. Upon implementation of these feature vectors on a sample of our dataset, we observed that they did not perform as well as expected in terms of reducing the dimensionality of the images. For instance, in the case of the shoes shown in Figure 3 below, a substantial amount of detail in the shoe was picked up by the feature. While the feature extraction parameters can be tweaked to lessen this effect, we found that we could not find consistent parameters that performed well enough across our different dataset images and categories. We therefore decided to exclude these features from our classification modeling.

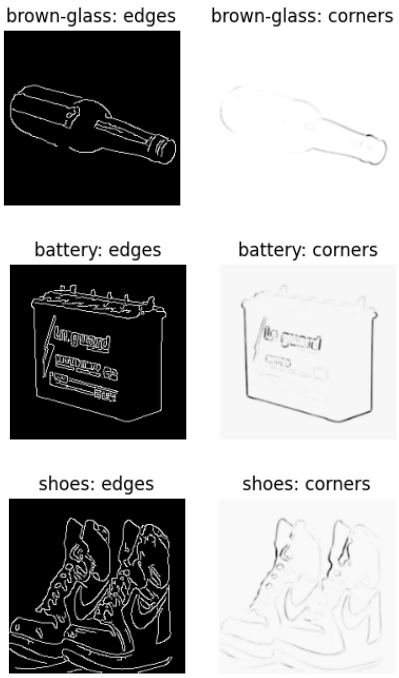


Figure 3: Sample Edge and Corner Feature Extraction

3.2 HISTOGRAM OF GRADIENTS (HOG)

We next explored HOG feature extraction. Unlike edge and corner detection, the HOG descriptor segments an image into sections and extracts image edges within those segments, while preserving both edge magnitude and orientation. Figure 4 below shows a sample HOG feature extraction on the same images shown in Figures 2 and 3.

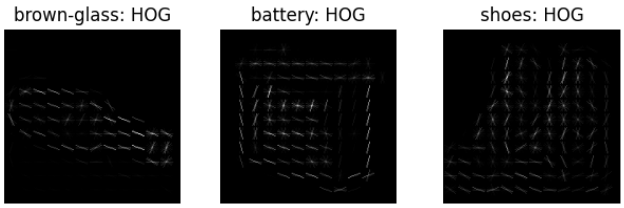


Figure 4: Sample HOG Feature Extraction

3.3 BAG OF VISUAL WORDS (BOVW)

The BOVW model is inspired by the bag of words model that is typically used as a basic natural language processing model. The BOVW model involves using an extractor, in our case the OpenCV Scale Invariant Feature Transform (SIFT) extractor[2], to identify image keypoints and descriptors. Keypoints are corners of an image that are identified in a manner robust to scale variations, while descriptors are patches of the image found around a keypoint. We expected this robustness to scale variations to help with identifying objects in our images that are of varying scale and size. These "visual words" are extracted from each image, and then clustered in a "bag", or codebook, using K-means clustering. The visual words from each image are then compared to the resulting codebook in order to identify matches. Figure 5 shows the keypoints and descriptors implemented on a sample of dataset images.

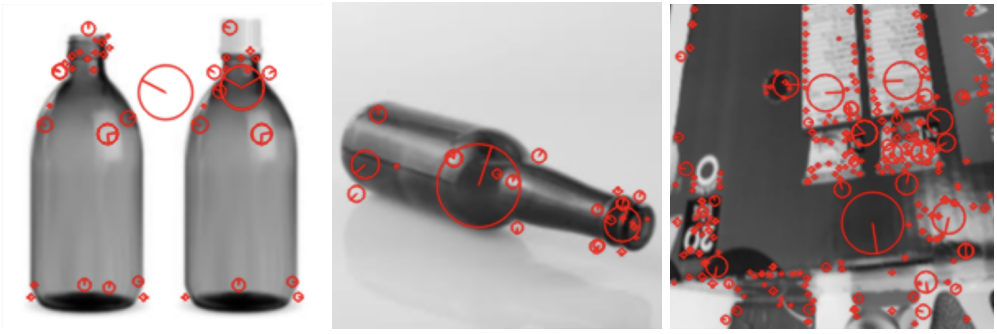


Figure 5: Sample of BOVW Keypoints and Extractors

3.4 BAG OF VISUAL COLORS (BOVC)

The BOVC model functions similarly to the BOVW model, where instead of keypoints and descriptors being extracted as image "words", the image's dominant colors are extracted and clustered into a color "bag", or codebook. In this case, K-means clustering is utilized in two steps. First, on every image to extract the [2]dominant colors, and then to cluster the images into the codebook. We chose to explore this feature due to its potential impact on differentiating brown, green, and white glass images. Figure 6 shows the top 3 colors extracted from a sample image of each of the three aforementioned classes.

---

[2] Introduction to the SIFT extractor: https://docs.opencv.org/4.x/da/df5/tutorial_py_sift_intro.html

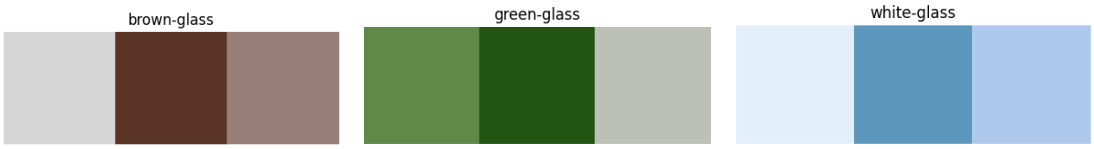


Figure 6: Dominant Three Colors in a Brown, Green, and White Glass Image using K-Means

3.5 RESIDUAL NETWORK (RESNET)

ResNet50 is a pre-trained neural network model that is able to handle diverse datasets through residual learning. We chose this method as it is able to work with images with objects of varying sizes and positions within an image, which would be a common situation when it comes to trash sorting and the uncertainty of what is being sent through the recycling plant. The last average pooling layer of ResNet50 averages the feature map across the image, creating a strong feature extractor that can account for the higher level complex information about each image.

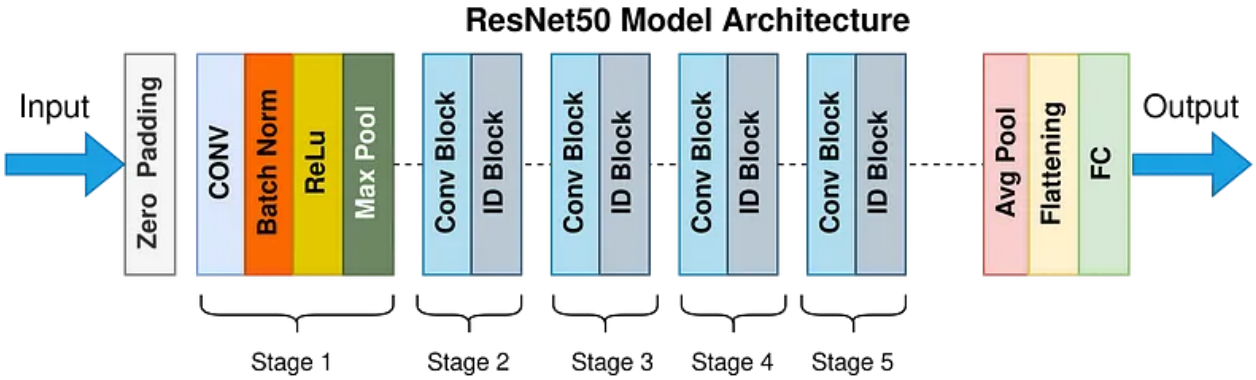


Figure 6: ResNet50 Model Architecture

4. DIMENSIONALITY REDUCTION

We implemented dimensionality reduction to improve the efficiency of our feature extraction methods, as well as tackle the curse of dimensionality. Edge and corner detection were unable to reduce dimensionality sufficiently, which is a given due to the nature of their detail-oriented approach to each edge and corner. HOG features also tend to have high dimensionality, so we applied PCA to help with dimensionality reduction. We then implemented t-SNE to visualize HOG, BOVW, BOVC, and ResNet to further understand the features and their usefulness.

4.1 PRINCIPAL COMPONENT ANALYSIS (PCA)

PCA is a technique that reduces the dimensionality of a dataset by linearly transforming it into a lower dimension. Implemented correctly, this method preserves most of the information of the dataset while greatly reducing computational complexity. While there is some loss of some information due to the transformation, this can actually aid in classification by preventing overfitting and making the dataset more generalizable.

Upon excluding edge and corner feature extraction from our classification, we implemented PCA on our HOG feature vector only. This is because both BOVW and BOVC have substantial dimensionality reduction already included in their algorithms through K-mean clustering techniques used to specify the number of visual words or colors in the codebook.

Figure 7 plots the explained variance with respect to the number of PCA components chosen for the HOG feature vector. Our goal was to choose the number of components that reduced dimensionality while still having an explained variance of roughly 90%, which resulted in 450 components chosen. For the remainder of this paper, references to HOG PCA will assume 450 components.

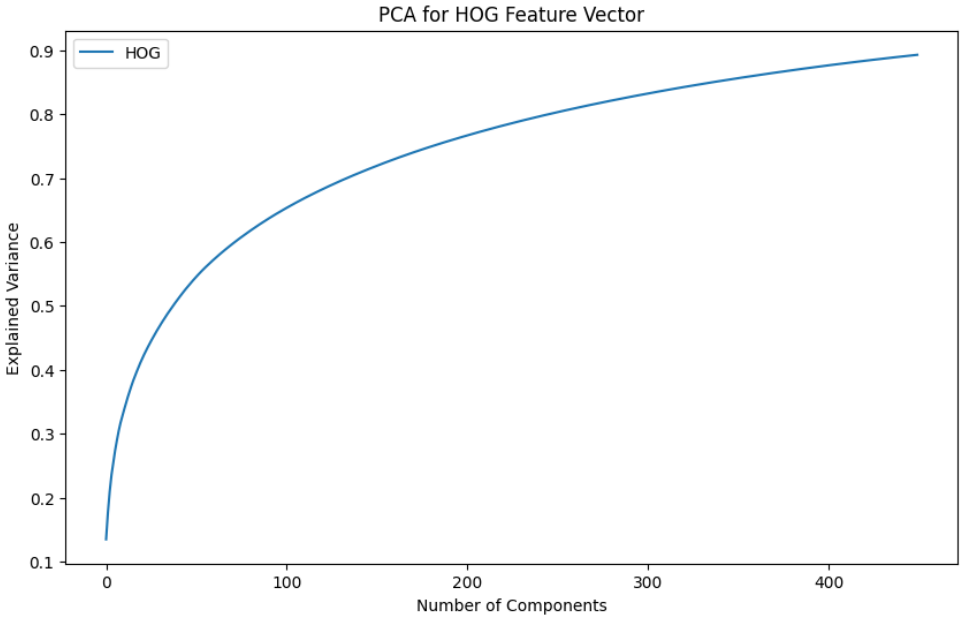
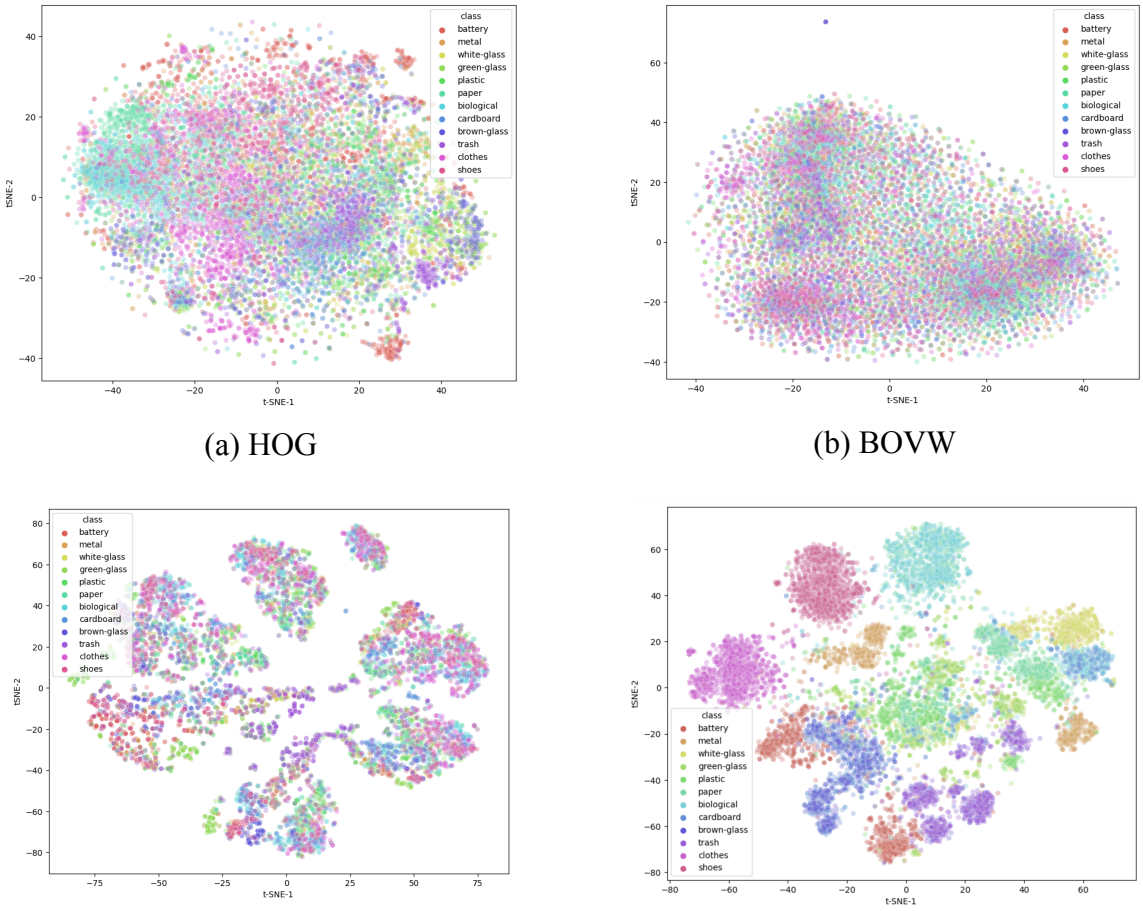


Figure 7: Explained Variance for HOG Feature Vector PCA

4.2 t-DISTRIBUTED STOCHASTIC NEIGHBOR EMBEDDING (t-SNE)

t-SNE is a dimensionality reduction technique that is specifically useful for visualizing high dimensional data. It provides insight into how well feature vectors cluster the different classes of a dataset. An ideal t-SNE visualization plot shows data points belonging to the same class clustered together. Figure 8 below shows the t-SNE visualization results for HOG, BOVW, BOVC, and ResNet. ResNet is the only feature vector shown where class clustering is truly achieved. It is important to note, however, that while a good t-SNE result indicates that the feature is likely favorable for classification, an unfavorable result does not necessarily indicate that the feature will not yield a good classification result.



(a) HOG (b) BOVW

(c) BOVC (d) ResNet

Figure 8: t-SNE Visualization

5. CLASSIFICATION

We explored two different types of models to accomplish the goal of classifying the type of garbage present in our images: Logistic Regression and a simple Perceptron. Our classification strategy was to use logistic regression to experiment with each of our feature vectors individually, concatenate better performing features, and select the top performing models to utilize with simple perceptron, conduct hyperparameter testing, and perform final predictions using our test set. Appendix 9.2 shows the confusion matrices associated with each of our preliminary logistic regression classification trials.

5.1 LOGISTIC REGRESSION

We decided to explore this linear classification algorithm due to its simplicity and explainability. This would help us determine whether or not a linear relationship can be used to differentiate between classes in our garbage dataset. The limitation to this approach is that it assumes linearity, and if this is not the case our model will suffer from this assumption. Many of our

feature vector classification experimentation resulted in overfitted results, as can be seen in the confusion matrices in Appendix 9.2. While we attempted to combat this through both feature vector and model parameter tuning, time constraints prevented us from achieving any substantial improvements to overfitted models. It is also interesting to note that in our ResNet classification confusion matrices, clothes and shoes classes appear to perform far better than other classes, which could be an indication of the appearance of items belonging to these classes in the pre-training of the model.

The following figure shows the training and validation confusion matrices our best performing logistic regression model with ResNet, HOG, and BOVC, which resulted in a validation accuracy of 82%:

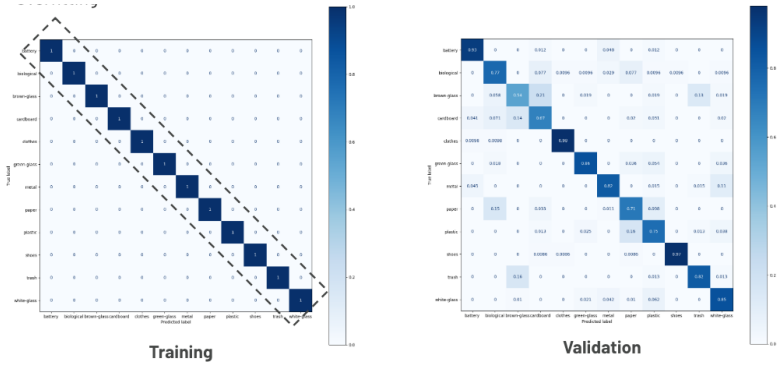


Figure 8: Logistic regression confusion matrix with HOG, BOVC, and ResNet

5.1A Hyperparameter Search

We performed a hyperparameter search using SKLearn's GridSearchCV package to optimize for the following parameters:

- 'solver': lbfgs, liblinear, sag, saga
- 'multi_class': auto, ovr, multinomial
- 'fit_intercept': True, False

Our results yielded these optimal parameters: **'solver': saga, 'multi_class': auto, 'fit_intercept': True**. Utilizing these parameters, we obtained a test set accuracy of 83% with our best performing logistic regression model.

5.2 SIMPLE PERCEPTRON

We wanted to utilize a basic neural network model to explore the potential of a more complex approach to classifying objects. This model does assume linearity similarly to logistic regression, which remains a limitation in the case the data is not linearly separable. However, this model is more capable of handling non-linear decision boundaries due to its flexibility as a neural network. This model can incorporate more complex features, which could potentially improve its classification performance.

The following figure shows the training and validation confusion matrices of our best performing perceptron model, with ResNet, HOG, and BOVC, which resulted in a validation accuracy of 85%:

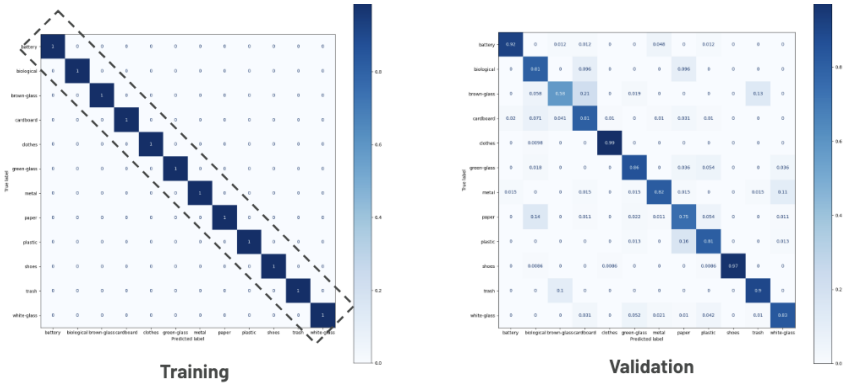


Figure 9: Perceptron confusion matrix with HOG, BOVC, and ResNet

5.2A Hyperparameter Search

We performed a hyperparameter search using SKLearn's GridSearchCV package to optimize for the following parameters:

- 'hidden_layer_sizes' : [(100,), (100,100), (200,100), (200,100,100)]
- 'learning_rate': [.0001, .001, .01]
- 'activation': [identity, tanh, relu]

Our results yielded these optimal parameters: '**hidden_layer_sizes': (200,100), 'learning_rate': .0001, 'activation': tanh**. Utilizing these parameters, we obtained a test set accuracy of 85% with our best performing simple perceptron model, performing better overall than our logistic regression equivalent.

Model	Feature Vectors Included	Best Performing Parameters	Test Set Accuracy
Logistic Regression	ResNet, HOG, BOVC	solver: saga multi_class: autofit_intercept: True	83%
Simple Perceptron	ResNet, HOG, BOVC	hidden_layer_sizes: (200,100) learning_rate: .0001 activation: tanh	85%

Figure 10: Summary of best performing models for Logistic Regression and Simple Perceptron

6. GENERALIZABILITY

We split our dataset into train (70%, 7,149 images), validation (10%, 1,021 images), and test (20%, 2,043 images) sets before training our classifier. When we performed a hyperparameter search using parts of our validation set, the accuracy results were 85% for perceptron, and 82% for logistic regression. After this hyperparameter search, our model test accuracy results were 85% for perceptron, and 83% for logistic regression. Ultimately, we were unable to completely avoid overfitting our data. Due to the decomposing nature of trash images, it is difficult to effectively prepare a model to handle these images of garbage in varying states. Some objects, like batteries and shoes, are able to maintain shape, while others like cardboard and glass break down easily. Due to the variability, some features tend to perform better than others in certain conditions. We were not able to achieve complete generalizability, yet we were able to maintain some generalizability if the trash is photographed in an ideal environment with as many noticeable identifiers as possible.

7. EFFICIENCY vs ACCURACY

Recycling relies on accurate and efficient sorting to reduce contamination and improve their recycling methods. However, more accuracy requires more complex algorithms that take up more resources, which could end up slowing down the sorting process. Efficiency is crucial to ensuring that each batch of sorted recycling is free of contamination that could render the method of recycling ineffective. We explore the trade offs between efficiency and accuracy by optimizing a model for each, and determining which would be a better fit for our task.

7.1 EFFICIENCY

Our best efficiency-optimizing model was a simple perceptron using HOG and ResNet50. The accuracy of this model was 76%, the precision was 75%, and recall was 74%. HOG was able to handle local information on edges and textures by capturing gradients in each image. ResNet50 was able to add hierarchical information to the HOG results, creating a computationally efficient model with minimized complexity.

7.2 ACCURACY

Our best accuracy-optimizing model was a simple perceptron using HOG, ResNet50, and BOVC. The accuracy of this model was 85%, the precision was 85%, and the recall was 84%. By including BOVC we were able to add more information regarding the relationships and patterns of each image, which helped the model further differentiate between classes. While adding BOVC to the model increased the computational costs, it was able to raise accuracy, precision, and recall each by about 10%. This substantial improvement makes the results, in our opinion, worth the additional costs.

8. CONCLUSION

We set out to see if computer vision could assist recycling plants in the complex task of classifying and sorting waste. There are some inherent challenges in this task, as trash can often

be damaged or altered to a point where it is hard to recognize by sight, especially when it is surrounded by other trash as well. Recycling plants deal with a high volume of waste, and need to sort it efficiently so that the methods of recycling are applied to the correct materials.

If we had more time, our next steps would include further investigation of misclassified images to better understand where the model ran into difficulties, as well as identify any visual trends among the image classes.

In our project, we became familiar with the complexity of sorting waste. Some forms of trash are easily classifiable at sight, while some are far harder to recognize. Due to the high volumes of waste that pass through recycling plants, and the speed at which they sort trash to optimize costs, we believe that the best use of our project would be to aid human sorters. Having a model classify each object with varying confidence, and then having a human determine the class on those less-obvious objects, would improve the workload of recycling plants at scale.

9.1 APPENDIX: DATASET SAMPLE IMAGES

Batteries

Metal

Biological

Paper

Brown-glass

Plastic

Cardboard

Shoes

Clothes

Trash

Green-glass

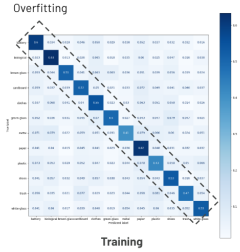
White-glass

9.2 APPENDIX: LOGISTIC REGRESSION CONFUSION MATRICES

Logistic Regression | BOVW

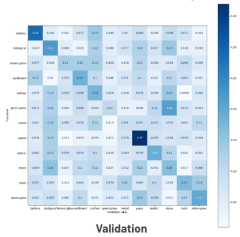
Words per Image=20, Codebook=100

Overfitting



Training

Validation Accuracy = 20%

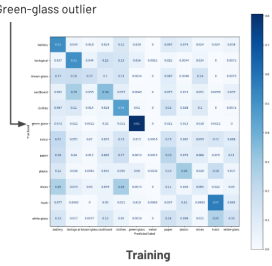


Validation

Logistic Regression | BOVC

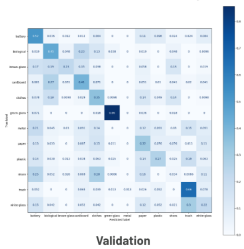
Colors per Image=3, Codebook=75

Green-glass outlier



Training

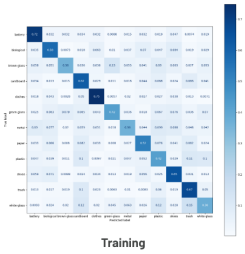
Validation Accuracy = 34%



Validation

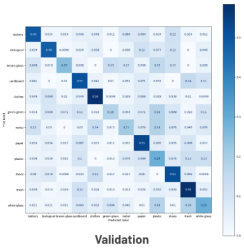
Logistic Regression | HOG

PCA 450 Components



Training

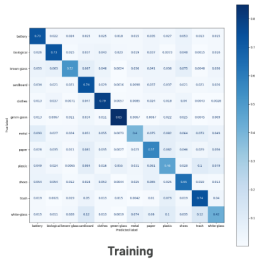
Validation Accuracy = 43%



Validation

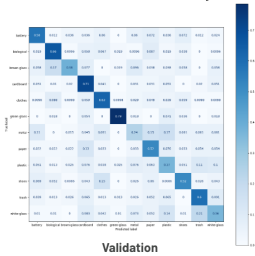
Logistic Regression | HOG + BOVC

HOG PCA 450 Components; BOVC Colors per Image=3, Codebook=75



Training

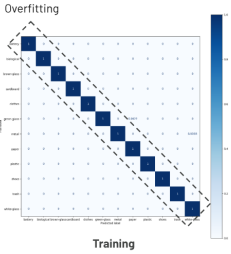
Validation Accuracy = 54%



Validation

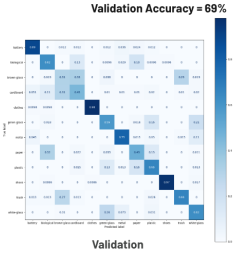
Logistic Regression | ResNet

Overfitting



Training

Validation Accuracy = 69%

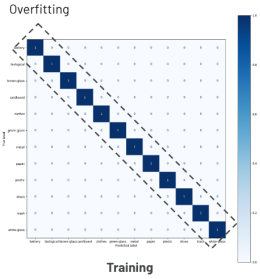


Validation

Logistic Regression | HOG + ResNet

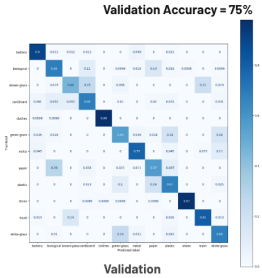
HOG PCA 450 Components

Overfitting



Training

Validation Accuracy = 75%

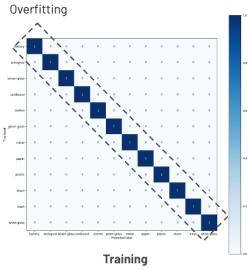


Validation

Logistic Regression | HOG + BOVC + ResNet

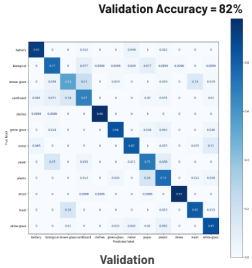
HOG PCA 450 Components; BOVC Colors per Image=3, Codebook=75

Overfitting



Training

Validation Accuracy = 82%



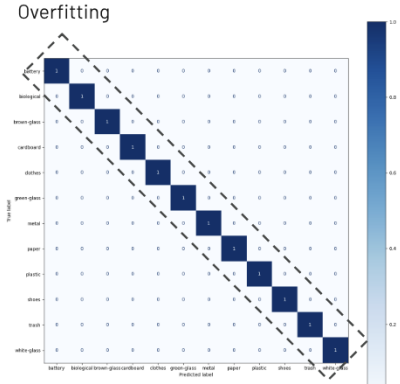
Validation

9.3 APPENDIX: PERCEPTRON CONFUSION MATRICES

Perceptron | HOG + BOVC + ResNet

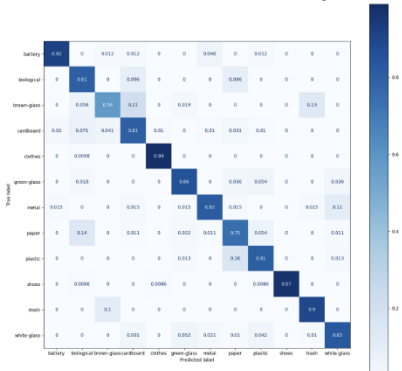
HOG PCA 450 Components; BOVC Colors per Image=3, Codebook=75

Overfitting



Training

Validation Accuracy = 85%

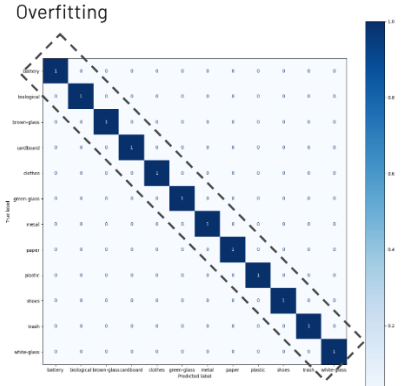


Validation

Perceptron | HOG + ResNet

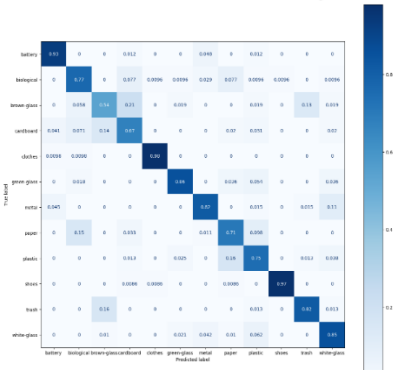
HOG PCA 450 Components

Overfitting



Training

Validation Accuracy = 76%



Validation