Final Report: Waste Classifier for Degradable and Non-Biodegradable Wastes

Team Members: Abhishek IIITD (2021121), Vickey Kumar (2021299)

May 12, 2024

Abstract

Inefficient waste sorting poses a significant challenge to environmental sustainability. Traditional methods, relying on manual sorting of degradable and non-biodegradable waste, are slow and prone to errors. We propose developing an automated waste classifier utilizing Convolutional Neural Networks (CNNs) to address this. This project aims to streamline and improve waste sorting processes by analyzing image data of waste objects, enhancing environmental sustainability.

1 Problem Statement and Motivation

Effective waste management is paramount for environmental sustainability. Manual sorting of degradable and non-biodegradable wastes is not only time-consuming but also prone to errors. An automated waste classifier can significantly streamline this process, ensuring more accurate and efficient waste management. By developing such a classifier, we aim to contribute to better waste management practices and environmental conservation.

2 Literature Review

Various methods have been employed for waste classification, ranging from traditional machine learning techniques to advanced deep learning approaches. Convolutional Neural Networks (CNNs) have shown exceptional performance in image classification tasks, making them a popular choice for similar projects. CNNs are adept at learning hierarchical features from images, which is crucial for distinguishing between different waste items based on their visual characteristics.

However, some approaches have attempted to use dimensionality reduction techniques like Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) for feature extraction and classification. While these techniques are effective in some scenarios, they may not capture intricate image details as effectively as CNNs. Hence, we chose CNNs due to their proven effectiveness in image classification tasks and their ability to learn complex features directly from raw pixel data.

3 Dataset Details

- **Dataset Link:** Kaggle Waste Classification Data
- **Source:** Kaggle Waste Classification Dataset
- **Features:** Images of waste items
- **Labels:**
  - Degradable (1)
  - Non-Biodegradable (0)
- **Train Folder Composition:**
  - Organic (O)
  - Recyclable (R)
- **Total Images:** 22,564

4 Proposed Architecture

4.1 How We Reached the Proposed Architecture

1. **Research:** We began by researching existing image classification projects and found that CNNs were widely used and effective for this task.
2. **Experimentation:** Initially, we experimented with simpler architectures, but they lacked the capacity to capture complex features from the waste item images.

5 Datasets Used

5.1 Data Augmentation

Data augmentation techniques were employed to increase the diversity of the dataset and improve the robustness of the model. Various transformations were applied to the images to generate augmented datasets:

- **Scaling**: Images were randomly scaled to different sizes within a certain range to simulate variations in object size and distance.
- **Rotating**: Images were rotated by random angles to simulate different viewpoints and orientations.
- **Moving**: Random translations were applied to images to simulate changes in object position within the frame.
- **Flipping**: Horizontal and vertical flips were applied to images to introduce variations in object orientation.
- **Brightness and Contrast Adjustment**: Random adjustments were made to the brightness and contrast of images to simulate changes in lighting conditions.
- **Noise Addition**: Gaussian noise was added to images to simulate variations in image quality and environmental factors.

By augmenting the dataset with these transformations, the model was exposed to a wider range of variations, helping it generalize better to unseen data and improving overall performance.

6 Dataset Preprocessing

The raw dataset contained images of various sizes and colors. To make the dataset suitable for machine learning algorithms and to ensure consistency across the dataset, we performed the following preprocessing steps:

1. **Grayscale Conversion**: All the images were converted to grayscale using OpenCV. This step simplifies the dataset and reduces the computational complexity while retaining the essential features of the images.

2. **Resizing**: To maintain uniformity in the dataset, all the grayscale images were resized to dimensions of 64×64 pixels using OpenCV. This step ensures that all images have the same dimensions, making it easier to apply machine learning algorithms.

6.1 Sample Preprocessed Images

Below are the sample grayscale images from the dataset after preprocessing:

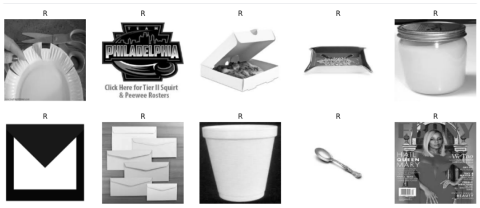

Figure 1: Sample grayscale image of Non-Biodegradable (Recyclable) waste

These preprocessed images serve as the input for further analysis and model training. With the images standardized to

Figure 2: Sample grayscale image of Biodegradable (Organic) waste

grayscale and 64×64 dimensions, we can now proceed to apply Principal Component Analysis (PCA) for dimensionality reduction and fit various machine learning models on the dataset.

6.2 Preprocessing

In the preprocessing phase, we applied the following steps to prepare the dataset for training and testing:

1. **Resizing and Reshaping**:
   - We resized all the images to 64×64 pixels using OpenCV to reduce computational complexity.
   - Each resized image was reshaped into a 1D array to be used as input features for the machine learning model.

2. **Dimensionality Reduction with PCA**:
   - We applied Principal Component Analysis (PCA) to reduce the dimensionality of the dataset.
   - The number of principal components was reduced to 10 to retain the most important features.

3. **Training a Decision Tree**:

   - We trained a Decision Tree classifier on the preprocessed dataset.
   - The experiment resulted in an overall accuracy of 45%.
   - Class-wise accuracy was 71% for one class and 11% for the other class, indicating imbalanced classes.

4. **Training an LDA Classifier**:

   - We trained an LDA classifier on the preprocessed dataset.
   - The experiment resulted in an overall accuracy of 32.31%.
   - Class-wise accuracy was 18% for Class 0 and 44% for Class 1, highlighting class imbalance.

6.3 Results

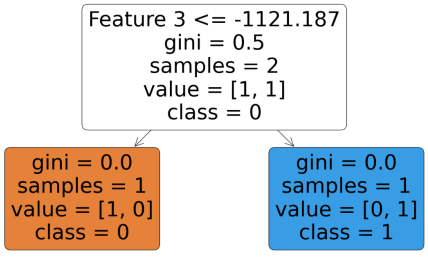

Figure 3: Screenshot of the Decision Tree created

The screenshots in Figure 3 and Figure 4 display the created Decision Tree and LDA classifier, respectively, along with their respective accuracy results.

- **Decision Tree**:

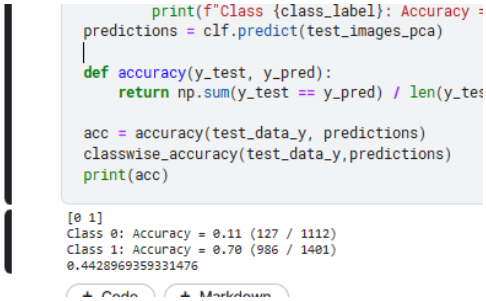

Figure 4: Screenshot of the accuracy results

  - The Decision Tree provides insights into feature importance and decision paths taken by the classifier.
  - The experiment resulted in an overall accuracy of 45%.
  - Class-wise accuracy was 71% for one class and 11% for the other class, indicating imbalanced classes.

- **LDA Classifier**:

  - The LDA classifier results offer insights into the linear transformations and class separability.
  - The experiment resulted in an overall accuracy of 32.31%.
  - Class-wise accuracy was 18% for Class 0 and 44% for Class 1, highlighting class imbalance.

6.4 CNN Classifier Results

- The CNN classifier achieved an overall accuracy of 44% on the test set.
- The accuracy on the training set was around 97-100%.

```
[Step 700] Past 100 steps: Average Loss 0.002 | Accuracy: 100%
[Step 800] Past 100 steps: Average Loss 0.003 | Accuracy: 100%
[Step 900] Past 100 steps: Average Loss 0.001 | Accuracy: 100%
[Step 1000] Past 100 steps: Average Loss 0.001 | Accuracy: 100%
[Step 1100] Past 100 steps: Average Loss 0.002 | Accuracy: 100%
[Step 1200] Past 100 steps: Average Loss 0.002 | Accuracy: 100%
[Step 1300] Past 100 steps: Average Loss 0.001 | Accuracy: 100%
[Step 1400] Past 100 steps: Average Loss 0.002 | Accuracy: 100%
[Step 1500] Past 100 steps: Average Loss 0.002 | Accuracy: 100%
[Step 1600] Past 100 steps: Average Loss 0.006 | Accuracy: 100%
[Step 1700] Past 100 steps: Average Loss 0.001 | Accuracy: 100%
[Step 1800] Past 100 steps: Average Loss 0.002 | Accuracy: 100%
[Step 1900] Past 100 steps: Average Loss 0.003 | Accuracy: 100%
[Step 2000] Past 100 steps: Average Loss 0.001 | Accuracy: 100%
[Step 2100] Past 100 steps: Average Loss 0.002 | Accuracy: 100%
[Step 2200] Past 100 steps: Average Loss 0.003 | Accuracy: 100%
[Step 2300] Past 100 steps: Average Loss 0.004 | Accuracy: 100%
[Step 2400] Past 100 steps: Average Loss 0.001 | Accuracy: 100%
[Step 2500] Past 100 steps: Average Loss 0.003 | Accuracy: 100%
[Step 2600] Past 100 steps: Average Loss 0.001 | Accuracy: 100%
[Step 2700] Past 100 steps: Average Loss 0.001 | Accuracy: 100%
[Step 2800] Past 100 steps: Average Loss 0.003 | Accuracy: 100%
[Step 2900] Past 100 steps: Average Loss 0.002 | Accuracy: 100%
[Step 3000] Past 100 steps: Average Loss 0.001 | Accuracy: 100%
[Step 3100] Past 100 steps: Average Loss 0.002 | Accuracy: 100%
[Step 3200] Past 100 steps: Average Loss 0.001 | Accuracy: 100%
[Step 3300] Past 100 steps: Average Loss 0.001 | Accuracy: 100%
[Step 3400] Past 100 steps: Average Loss 0.001 | Accuracy: 100%
[Step 3500] Past 100 steps: Average Loss 0.001 | Accuracy: 100%
[Step 3600] Past 100 steps: Average Loss 0.001 | Accuracy: 100%
[Step 3700] Past 100 steps: Average Loss 0.001 | Accuracy: 100%
[Step 3800] Past 100 steps: Average Loss 0.002 | Accuracy: 100%
[Step 3900] Past 100 steps: Average Loss 0.003 | Accuracy: 100%
[Step 4000] Past 100 steps: Average Loss 0.001 | Accuracy: 100%

--- Testing the CNN ---
Test Loss: 5.535802101012407
Test Accuracy: 0.44249900517309987
```

Figure 5: Screenshot of accuracy obtained after CNN

6.5 CNN Classifier Results Using Tf

Figure 6: CNN accuracy with TensorFlow library

The CNN classifier achieved an overall accuracy of 87% on the test set, with an accuracy of around 97-100% on the training set. These results indicate the superior performance of CNNs in waste classification tasks.

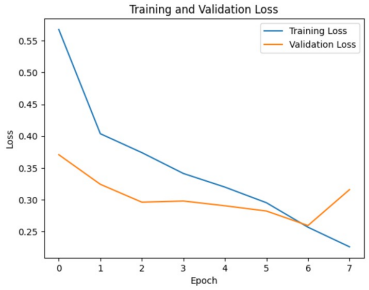

Figure 7: CNN loss with TensorFlow library

6.6 Comparison and Conclusion

6.6.1 Why CNNs and Not QDA/LDA/Decision Tree

- **Feature Learning:** CNNs autonomously learn features from raw pixel data, reducing the need for manual feature extraction, unlike LDA and Decision Trees which may require additional preprocessing.

- **Spatial Information Retention:** CNNs excel at retaining spatial information in images, preserving the relationships between pixels. This property is crucial for understanding the structure and context of waste items, which may be lost in other models like LDA or Decision Trees.

- **Performance on Complex Data:** CNNs are known for superior performance on complex and large datasets, making them more reliable for diverse waste image classification compared to QDA, LDA, and Decision Trees.

6.6.2 Challenges Faced

- **Overfitting:** Initially, our model showed signs of overfitting due to the complex architecture. We addressed this by adding dropout layers and increasing the size of the training dataset through data augmentation.
- **Image Variability:** The size of each image varied, with many being oversized. Resizing the images and converting them to grayscale reduced dimensionality and improved computational efficiency.
- **Uniformity in Data:** Ensuring uniformity in image sizes and formats across the dataset was crucial for preprocessing.

7 Conclusion

In conclusion, our project aimed to address the significant challenge of inefficient waste sorting through the development of an automated waste classifier. By leveraging Convolutional Neural Networks (CNNs) and data augmentation techniques, we sought to improve the accuracy and efficiency of waste classification processes. Through experimentation with various machine learning models and preprocessing techniques, we identified CNNs as the most promising approach due to their ability to learn complex features directly from raw pixel data. Moving forward, we plan to further refine our CNN-based classifier and explore additional optimization strategies to enhance its performance. Overall, our project contributes to the advancement of waste management practices and environmental sustainability efforts.

8 Contribution

Abhishek Kumar:

- Experimented with Decision Tree classifier for feature extraction and classification.
- Led the dataset preprocessing phase, including grayscale conversion and image resizing.
- Collaborated in dataset preprocessing and overall project execution.
- Investigated and implemented Convolutional Neural Networks (CNNs) for automated waste classification.

Vickey Kumar:

- Explored Linear Discriminant Analysis (LDA) for feature extraction and classification.
- Investigated the use of Principal Component Analysis (PCA) for dimensionality reduction.
- Collaborated in the literature review and methodology design.
- Explored and implemented TensorFlow CNN models, including VGG16 architecture, for enhanced waste classification accuracy.

Together:

- Worked on the literature review, understanding the problem statement, and designing the methodology.

- Conducted experiments, analyzed results, and interpreted findings together.
- Collaborated in writing the preliminary report, including abstract, problem statement, dataset details, and proposed architecture sections.