Material Classification for Embedded Systems

Chaitanya A.
Department of Computer Science
DHBW Stuttgart
inf22029@lehre.dhbw-stuttgart.de

Lukas K.
Department of Computer Science
DHBW Stuttgart
inf22150@lehre.dhbw-stuttgart.de

Abstract

It is clear that environmental concerns, particularly waste management and recycling, are pressing issues worldwide. At the same time, advancements in artificial intelligence (AI) have opened up new possibilities for addressing these challenges. This project explores the integration of AI into waste management through a convolutional neural network (CNN)-based waste classification system. Using the TrashNet dataset, which categorizes waste into six distinct types, the model aims to simplify and improve recycling efficiency.

The dataset was preprocessed to enhance model generalization, involving resizing, normalization, and augmentation. The CNN architecture incorporates multiple convolutional, pooling, and dense layers, with dropout to mitigate overfitting. Training utilised the Adam optimiser and a sparse categorical cross entropy loss function, with early stopping to prevent overtraining. While the model achieved a validation accuracy of 72.15%, falling short of the 80% target, this study unequivocally demonstrates the potential for AI in waste classification. Future work will explore alternative architectures, hyperparameter tuning, larger datasets and higher computational resources to improve accuracy and scalability.

1 Introduction

In recent years, there has been much discussion about environmental issues, including the importance of waste separation. The pollution of seas and the transportation of waste from developed countries to developing countries have become a significant concern for many individuals, businesses, and governments. Another subject that has attracted considerable attention in recent years is artificial intelligence (AI), which has become the most discussed topic in the technology sector. The central question guiding this study is the potential for integrating these two significant issues into a unified and pragmatic framework. The proposed solution, MAterial Classification, employs a convolutional neural network to analyze images of waste and categorize them into distinct categories, with the objective of simplifying recycling for different types of waste.

The goal is to reach an accuracy of at least 80% to ensure that the model is useful. Furthermore, the model should utilise a minimal number of parameters to ensure its compactness and thus enhance its functionality in embedded systems, which typically possess limited computing power. The project can be accessed on the GitHub platform via the following link: `https://github.com/Cha1tanyaa/Material-Classification.git`(14.06.2025)

2 Related work

Gary Thung and Mindy Yang have already done work on waste classification. In their 2016 paper [1], they compared the performance of a Convolutional Neural Network (CNN) to a Support Vector

Machine (SVM). At the time, no public datasets for waste classification were available, so they created their own [2]. Initially, the dataset consisted of approximately 2400 images, but has since grown to more than 5000 images.

The SVM model was trained using a 70/30 train-test split. For the CNN, they used a model structure similar to AlexNet. Their training process included a dataset split of 70/13/17 for training, validation and testing, respectively, with images resized to 256x256 pixels. Training was carried out over 60 epochs with a batch size of 32 and a learning rate of $5e-8$. The test accuracy achieved was 63% for the SVM and only 22% for the CNN. They noted difficulties in getting the CNN to learn effectively. However, the GitHub repository for their project mentions that the CNN accuracy has since been improved to 75%.

Their implementation was done in Lua, and the CNN utilized a pre-trained model similar to AlexNet, but with only three-fourths of its parameters. In contrast, our paper will employ Python and TensorFlow [3] to build and train CNNs, as these tools offer greater convenience. We will also design a custom model architecture to better address issues like overfitting and underfitting. This will also make it easier to adjust the model to increase performance.

3 Dataset

This project uses the TrashNet dataset from HuggingFace [2], which contains six distinct categories of waste:

- Paper
- Glass
- Plastic
- Cardboard
- Metal
- Trash

We used the datasets library [4] for downloading and managing the dataset. This dataset is particularly well-suited for the classification of waste due to its comprehensive representation of diverse materials. To illustrate the diversity of the dataset, below are a few sample images:

Table 1: Three of the original images

The dataset has been segmented into three categories for training, validation, and testing purposes, following a 60/20/20 split. Prior to this, the dataset was randomly shuffled and a fixed seed was utilised to ensure reproducibility. The dataset underwent several steps to ensure its suitability for machine learning. Initially, the images were found to be larger than the model's capacity, necessitating a resizing to 100x100 pixels. Subsequently, the pixel values were normalised through scaling to a range between 0 and 1. One such example is provided below.

Each image in the dataset was assigned a numerical label ranging from 1 to 6, corresponding to the six distinct waste categories.

Given the relatively small size of the dataset, the risk of overfitting was a significant challenge. To overcome this limitation and improve the generalisation capabilities of the model, data augmentation techniques were employed. This process involved generating additional variations of the existing data

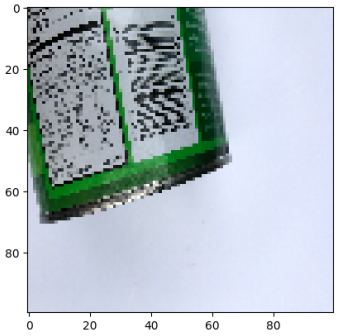

Figure 1: Downscaled Image

through transformations such as flipping images, rotating at different angles and adjusting brightness. The following illustration shows a augmented image:

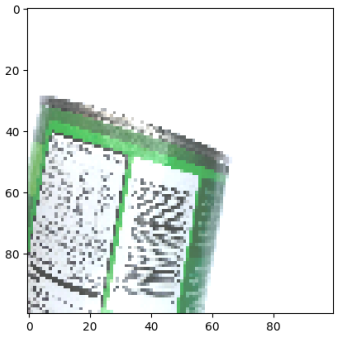

Figure 2: Augmented Image

Initially we had about 3000 training images, 1000 validation images and 1000 test images. But through data augmentation, the number of training images increased significantly to a total of about 15000.

4 Methods

This project uses Convolutional Neural Networks (CNNs) since the task at hand is image classification. The architecture of the CNN consists of multiple convolutional layers, max-pooling layers and fully connected layers. The model consists of six layers. A input layer with a shape of 100x100x3. These represent the height, width, and the three RGB color channels, four hidden layers and a softmax layer with 6 classes as output.

The model uses the sparse categorical loss function. The following shows the formula of the function:

$$L(y, \hat{y}) = -\sum_{i=1}^{C} y_i \log(\hat{y}_i)$$

This approach is frequently employed in multiclass classification tasks. In this context, C denotes the total number of classes, y_i denotes the true label indicator (1 for the correct class, 0 otherwise), and $\hat{y}_i$ represents the predicted probability for class i. This loss evaluates how well the predicted probabilities $\hat{y}$ match the true labels y, with lower values indicating better performance. [6]

The Adam optimizer is used to minimize the loss, which is an adaptive gradient-based optimization algorithm that adjusts the learning rate for each parameter. It combines the benefits of AdaGrad and RMSProp by updating the parameters using the first (mean) and the second moment (uncentered variance) of the gradients. The following formula is used to update the parameters:

$$\theta_t = \theta_{t-1} - \frac{\alpha}{\sqrt{v_t} + \epsilon} \cdot m_t$$

where m_t is the first moment estimate, v_t is the second moment estimate, α is the learning rate, and ϵ is a small constant for numerical stability. [7]

5 Results

The Early Stopping Hyperparameter was added into the model with a view to averting overfitting. This hyperparameter is responsible for monitoring the loss during training, and if the loss increases for a defined number of consecutive epochs, it halts the process. The purpose of this is to ensure that the training is halted at the optimal performance of the model, thus preventing the model from being subjected to prolonged training.

A learning rate of 0.001 was chosen to strike a balance between stability and efficiency during training. It is small enough to avoid overshooting the loss function, ensuring stable updates and gradual convergence to the global minimum. At the same time, it is large enough to allow reasonably fast progress without requiring excessive training epochs. In addition, this learning rate helps the model avoid getting stuck in plateaus or areas of the loss function with minimal gradient changes, allowing for consistent improvement over time.

The training process was executed over a period of 15 epochs, with a batch size of 100 resulting into 152 batches. Prior, the dataset was subjected to a randomization process to ensure diversified training, validation and test sets.

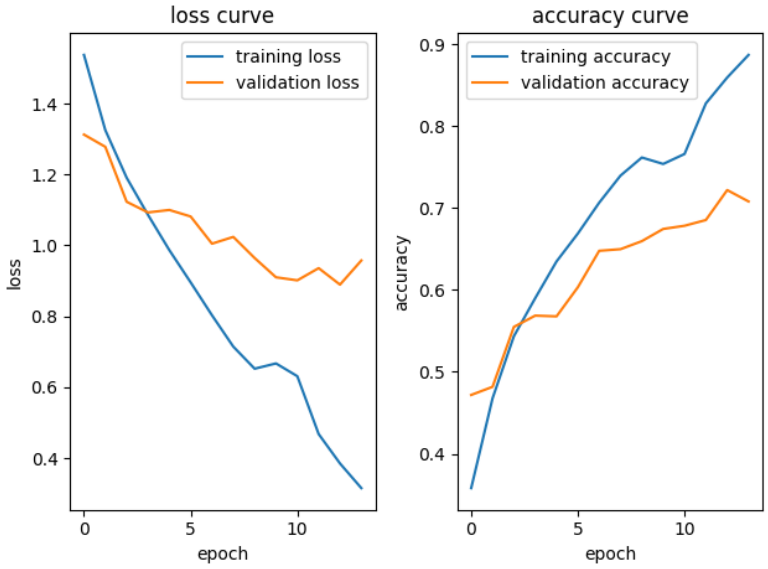

Figure 3: Loss curve and accuracy curve

As can be seen in Figure 3, the model training was stopped after the validation loss increased continuously for 2 consecutive epochs. Without the early stop hyperparameter, the model training would have continued and the loss would have been higher. As shown in the other diagram, the accuracy goes up. The validation accuracy reaches its highest point at around 70% before the training of the model is stopped. At the same time, the training accuracy reaches about 90%. This project utilized only accuracy metrics, as recall and precision were not relevant in this context.

The following table shows the structure of the model:

Layer	Output Shape	#Parameters
Input	100x100x3	0
Conv2D (3x3)	98x98x32	896
MaxPooling2D (f=2, s=2)	49x49x32	0
Conv2D (3x3)	47x47x64	18496
MaxPooling2D (f=2, s=2)	23x23x64	0
Conv2D (3x3)	21x21x128	73856
MaxPooling2D (f=2, s=2)	10x10x128	0
Flatten	12800	0
Dense	256	3277056
Dropout (P=0.2)	256	0
Dense	6	1542

Table 2: Caption

The model reached a validation loss of 0.9118 and a validation accuracy of 0.7215.

6 Conclusion and future work

Although the goal of 80% was not reached we could see the importance of having enough sample images in the dataset. This project also highlighted the advantage of data augmentation as it was one of the main reasons why we were able to achieve an accuracy of over 70%. Additionaly the objective was to utilise as few parameters as possible and to maintain the model's structure in a compact form for utilisation in embedded systems. This may also be why we could not achieve the goal of 80%.

The extension of the project may be facilitated by the use of alternative model structures and the use of additional hyperparameters, through which may result in a higher test accuracy. Furthermore, higher computational power could be utilized to increase the model's complexity, as the systems we used were unable to handle the load. Additionally, capturing more real images could expand the dataset's diversity and reduce reliance on augmented pictures.

References

[1] Mindy Y., Gary T. & (2016) Classification of Trash for Recyclability Status. Stanford University. URL: https://cs229.stanford.edu/proj2016/report/ThungYang-ClassificationOfTrashForRecyclabilityStatus-report.pdf (11.01.2025)

[2] Mindy Y. Gary T. & (2024) *garythung/trashnet · Datasets at Hugging Face.* Image dataset. URL: https://huggingface.co/datasets/garythung/trashnet (11.10.2025)

[3] tensorflow 2.12.0 (2024) Google. URL: https://www.tensorflow.org/ (11.10.2025)

[4] datasets 3.2.0 (2024) HuggingFace. URL: https://pypi.org/project/datasets/ (11.10.2025)

[5] keras 2.10.0 (2015) Google. URL: https://libraries.io/pypi/keras/2.10.0 (11.10.2025)

[6] Terven, J. R., Cordova-Esparza, D. M., Ramirez-Pedraza, A., Chavez-Urbiola, E. A., Romero-Gonzalez, J. A. & (2024). Loss functions and metrics in deep learning. Preprint. URL: https://doi.org/10.48550/arXiv.1412.6980 (11.01.2025)

[7] Diederik P. Kingma, Jimmy Ba & (2017) Adam: A Method for Stochastic Optimization. URL: https://doi.org/10.48550/arXiv.1412.6980 (11.01.2025)