Comparative Analysis of Convolutional Neural Network Models for Solid Waste Categorization

TAMUNO-OMIE J. ALALIBO[1], NKOLIKA O. NWAZOR[2]
[1]*Computer Engineering Department, Rivers State University, Port Harcourt, Rivers State, Nigeria*
[2]*Electrical/Electronic Engineering Department, University of Port Harcourt, Rivers State, Nigeria*

Abstract- This paper is on the comparative analysis of various convolutional neural network (CNN) models that could be used in the classification of solid waste into different categories for recycling. Effective solid waste classification is a crucial aspect of waste disposal and recycling processes to maintain a sustainable environment. In recent years, CNNs have emerged as a powerful tool for solid waste classification, thanks to their ability to learn features from images and classify objects accurately. In this paper, the MATLAB platform was used to train eight pre-trained CNN models for solid waste classification. The models are AlexNet, GoogleNet, ResNet18, MobileNetV2, ResNet50, ResNet101, EfficientNetB0 and InceptionV3. Each model was trained for 5 epochs, 7 epochs, and 10 epochs respectively. The TrashNet dataset was used. The dataset was split into 75% and 25% for the training set and validation set respectively. After training for 5 epochs, ResNet50 achieved validation accuracy of 90.95%, ResNet101 had 90.48%, while the rest achieved accuracies ranging from 64% to 89%. After training for 7 epochs, ResNet50 got an accuracy of 92.06%, ResNet101 and InceptionV3 both had an accuracy of 91.59%, while EfficientNetB0 improved to 90.63%. After 10 epochs, ResNet101 achieved an accuracy of 92.38%, ResNet50 got an accuracy of 92.02%, while EfficientNetB0 and InceptionV3 achieved an accuracy of 91.43%. From the results, ResNet-50 and ResNet-101 achieved higher validation accuracy. In real-world applications, these models can be used in smart bin waste collection, smart waste recycling, and waste management in general.

Indexed Terms- Convolutional Neural Network, Waste Image Classification, TrashNet, ResNet50, Recycling

I. INTRODUCTION

The production of waste is an inevitable consequence of population growth, urbanization, and economic development [1]. As countries undergo economic growth and globalization, individuals are exposed to a broader array of products and services, resulting in a rise in waste generation. According to [2], solid waste refers to solid-state materials that are produced from various activities, deemed unwanted, lacking usefulness, and subsequently discarded by individuals or society. Most of these discarded items are recyclable materials [1][2]. However, for recycling to take place, waste classification and sorting are important. Manual classification is still popular but time-consuming and often prone to error, Deep learning (DL) techniques like Convolutional Neural Networks (CNNs) can aid automated classification to accurately classify solid waste into defined categories.

CNNs are deep neural networks usually used for image classification. It can recognize unique features like edges, textures, and shapes [3]. CNN divides images into small pixel blocks or features using stacked layers and filters allowing for efficient processing and accurate results. CNNs differ from traditional machine learning algorithms such as SVM in learning complex features automatically during training. CNNs use artificial neural networks (ANNs) to learn spatial hierarchies of features and can identify and extract high-level features directly from raw data, eliminating the need for human supervision [3][4]. This allows CNNs to detect diverse features, ranging from low-level patterns to high-level patterns. CNNs are highly effective in various image and video recognition tasks, such as self-driving cars, image search, medical imaging, and image classification. They also show promise in accurately segregating waste materials, paving the way for effective waste management practices. Despite requiring more training data, DL models typically yield good results for deployment [5].

This article presents a comparative analysis of CNN models for solid waste classification. The experiment compares the performance of eight CNN models on the TrashNet dataset using MATLAB environment. The models were trained at different numbers of epochs to determine the effect of epochs on the accuracy. The performance results were presented using the confusion matrix. The accurate identification of solid waste items using CNN models can contribute toward targeted recycling processes, ensuring that materials are appropriately recycled or disposed of, reducing waste contamination, and promoting resource conservation.

In [6], TrashNet, an open-source dataset for recyclable waste images, was curated and used for waste classification. The dataset contains 2527 images from various waste classes, including paper, plastic, glass, cardboard, metal, and general trash. The researchers applied SVM and CNN models to classify waste into eight categories. The AlexNet-like CNN model achieved 75% accuracy, while the SVM method achieved 63%. In [7], the CNN model was used as part of a waste management system to classify digestible and indigestible waste. The dataset was divided into an 80:20 ratio for the training set and validation set. The CNN model was trained for epoch lengths of 20, 27, and 35, respectively. The CNN model achieved a waste classification accuracy of 95.3125% for an epoch length of 35.

In [8], CNN models were compared for waste classification using the TrashNet dataset. The study compared the performance of ResNet50, ResNet18, and VGG16 CNN models for image classification on the TrashNet dataset. ResNet18 achieved the highest validation accuracy 87.8% through fine-tuning. In [9], CNN and SVM were used to classify waste into paper, plastic, and metal categories. The images were from the TrashNet dataset and the Internet. The study achieved accuracies of 82.2% and 79.4%, respectively. The focus was to use machine learning to automate industrial waste sorting, improving recycling effectiveness.

In [10], CNN models were used to classify nine waste categories. The dataset, divided in a ratio of 70: 30, was a combination of TrashNet and internet images. The models achieved waste classification accuracy between 91.9 to 94.6%. MobileNetV3 had a high classification accuracy (94.26%), small storage size (49.5 MB), and fast running time (261.7 ms). In [11], four CNN models were compared for waste segregation, with a focus on dry, wet, recycled, electronic, and medical categories. After training for 50 epochs, MobileNetV2 demonstrated the best results in detecting, predicting, and classifying waste objects using the CNN algorithm.

CNNs have shown promising results in image classification tasks, making them an attractive choice for automated solid waste classification. But the foundation of CNN is based on the principles of Artificial Neural Network (ANN).

II. OVERVIEW OF ARTIFICIAL NEURAL NETWORK

Artificial Neural Network (ANN) is a brain-inspired algorithm that is not specifically programmed, but it learns (or is trained) automatically and progressively through appropriate examples of input and output datasets [3][12]. ANN has nodes, called neurons, which are arranged in different layers. The neurons of the same layer do not connect, but they connect to neurons on other layers. Each connection has a weight value (W) that determines the influence of the neuron on the computation.

ANN has three types of layers namely input layer, hidden layer(s), and output layer [4]. The input layer accepts data into the ANN via input neurons. The hidden layers have neurons that process the input data to identify and learn patterns in the data presentation and then figure out the relationship between input data and output results [6]. Activation functions are used to add non-linearity to the output of the hidden layer's neuron, which helps in extracting complex features [12][13]. ANNs can have one to several hidden layers depending on the task being performed. The output layer gives the results and calculates the loss function of the ANN operation via output neurons [15]. Figure 1 shows a simple n-layer feedforward ANN. Equation 1 presents the output of each layer in ANN. Equations 2 – 5 present some common activation functions used in deep neural network models.

During training, deep neural networks experience vanishing gradient problem. This is when the gradient of the loss function is calculated and backpropagated toward the input layer to update its learnable parameters, weight (*W*), and bias (*b*) values [14][16]. Updating these parameters is how neural networks learn and it is an iterative process. The updating process is expressed in Equations 6 and 7. The goal is to reduce loss or error while improving the accuracy of the networks to perform specific tasks. Vanishing gradients can impede the learning process of deep neural networks [16]. The loss gradient gets smaller from one layer to another, when it is backpropagated toward the input layer. Thus, the weight of the earlier layers receives little or no update, which means the network cannot learn effectively during training [16][17]. Proper regularization and optimization of the network can reduce vanishing gradient issues. ANN-based models are used to solve classification tasks (i.e., binary class, and multi-class) and regression tasks through supervised and unsupervised learning techniques.

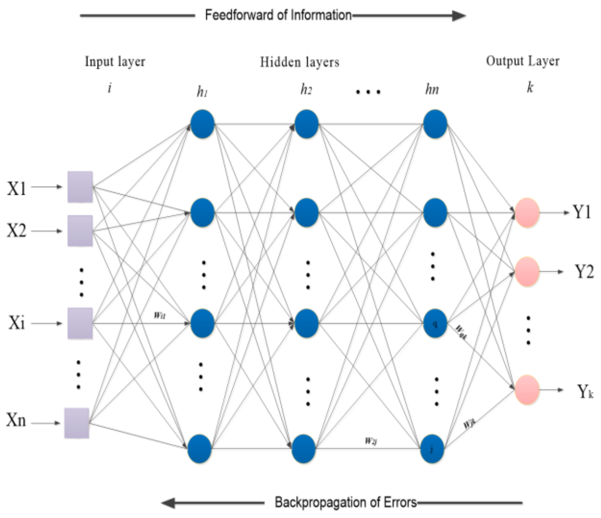

Figure 1: *n*-layer feedforward ANN

The output of layers:

$$Y_L = \text{Ꞁ}_L\left(b_L + \sum_{q,r=1}^{N}\left(W_{L,L-1}^{q,r} * h_{L-1}\right)\right) \quad (1)$$

Activation functions:

$$Z(x)_{sigmoid} = 1/(1 + e^{-x}) \quad (2)$$

$$Z(x)_{ReLu} = \max(0, x) = \begin{cases} x \text{ if } x > 0 \\ 0 \text{ if } x < 0 \end{cases} \quad (3)$$

$$Z(x)_{LeakyReLU} = \max(0.1x, x) \quad (4)$$

$$Z(x)_{softmax} = \exp(x_i)/\left(\sum_{j=1}^{K}\exp(x_j)\right) \quad (5)$$

Weight and bas values update:

$$W_{ij}^{t} = W_{ij}^{t-1} - \beta \cdot \frac{\partial Loss}{\partial W_{ij}^{t-1}} \quad (6)$$

$$b_l^{t} = b_l^{t-1} - \beta \cdot \frac{\partial Loss}{\partial b_l^{t-1}} \quad (7)$$

Where Y_L is the output of each layer *L*; $\hat{Y}_K$ is the final output of the network. Ꞁ(.) is the activation function; *L* is the layer of the network (*L* = 1, 2, …, *L-1*); $W_{L,L-1}^{q,r}$ is the weight value of *q*-th neuron on *L* and *r*-th neuron on *L-1* layers; *N* is the number of neurons on *L* layer; *K* is the number of classes; w_{ij}^{t} is the final weight in the current training epoch, w_{ij}^{t-1} is the preceding weight at (t-1) training epoch. *w* stands for each learnable parameter, *β* stands for a learning rate, and *E* stands for prediction error or loss function.

III. OVERVIEW OF CONVOLUTIONAL NEURAL NETWORKS

Convolutional Neural Networks (CNNs) are a type of deep neural network that is excellent for image classification tasks [16]. It is particularly effective in classifying images since it can recognize unique features within the images, such as edges, textures, and shapes. The key difference between deep learning techniques and traditional machine learning algorithms lies in the way deep learning algorithms learn unique and complex features within the data [15][16]. Traditional machine learning algorithms require engineers to manually identify every important feature within the data, while deep learning algorithms can automatically learn these features during the training process [15] [16]. Additionally, deep learning techniques require a larger amount of training data compared to traditional machine learning algorithms. Though DL models take more computational power, they normally produce good results for deployment.

Objects possess shapes that consist of lines, curves, and various other characteristics, which collectively contribute to their unique attributes. To identify these objects, CNNs leverage images represented as pixel values organized in a 2D or 3D grid, alongside small parameter grids known as kernels or filters [16][17]. These filters are feature extractors that can be optimized, enabling CNN to detect diverse features across shapes and edges present in the images. By utilizing multiple kernels or filters, CNNs extract a range of features from an image, generating multiple feature maps. The CNN architecture comprises different layers responsible for image classification and feature extraction. CNNs are trained using millions of images and can learn to recognize patterns and features within images. It has revolutionized computer vision and image recognition.

A. Layers of convolution neural network

CNNs consist of multiple layers of neurons where feature extraction and classification occur. CNN architectures are a set of pre-designed networks with different numbers of layers, filters, and feature extraction techniques that can be used for image classification tasks. The input *I* for each layer in a CNN model accepts data (image or feature maps) in a specific format presented in the form of (image height x image width x depth/channel number) or [$M_H \times M_W \times D$]. Each pixel value in the input matrix can range from 0 to 255. CNNs are capable of processing, detecting, and classifying images or objects within images, through key layers in their architecture. These layers are the convolution layer, pooling layer, and fully connected layer [17][18].

The convolutional layer employs filters to extract features from the input images and it is mathematically expressed in Equation 8. The pooling layers downsample the feature maps to reduce dimensionality. The fully connected layers perform the final classification based on the extracted features [15 – 17]. An activation function is applied to the output of each convolution and pooling layer in a CNN model. The activation function introduces non-linearity to the model, enabling it to learn complex features. To reduce the effect of vanishing gradient, regularization, and optimization techniques such as batch normalization, neuron dropout, and data augmentation are used [19]. Also, the ReLU activation function is commonly used because it helps prevent the issue of vanishing gradient [16]. Cross entropy, expressed in Equation 9, is the multi-class loss function used [17]. Figure 2 shows the structure of the Convolutional Neural Network (CNN).

$$y(i,j) = \text{A}_L\left(O_{i,j}\right) = \text{A}_L\left(\sum_q \sum_r k_{q,r} * m_{i+q,j+r}\right) \quad (8)$$

$$Loss\ (y,p) = -\sum_{c=1}^{M} y_{i,c}\ log\left(p_{i,c}\right) \quad (9)$$

Where $K_{m,n}$ is the filter matrix; $I_{i,j}$ is the input matrix; *A(.)* is the activation function (usually ReLU); $O_{i,j}$ denotes the feature map (that is, the element in the *i-th* row and the *j-th* column of the output matrix) computed using a convolution operation; *p* is the predicted class probability for a sample same as equation 5; $y_{i,c}$ is the true class with the binary label (0 or 1) if class label *c* is the correct classification for a sample *I*; *log* is the natural logarithm; e^{w_i} represents the non-normalized output from the preceding layer; *N* represents the number of neurons in the output layer; *M* is the number of classes.

Figure 2 shows the architecture of a typical CNN.

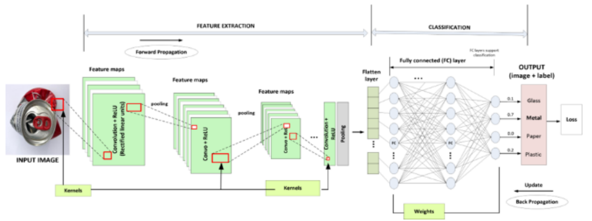

Figure 2: Architecture of a CNN

B. CNN models for solid waste classification

CNNs have proven to be highly effective in image classification tasks, and they have been applied to a variety of image recognition problems, including solid waste classification. CNNs analyze the images of waste items and learn the distinguishing features that differentiate them. Various architectures have been developed over the years by researchers to improve the accuracy and speed of image recognition.

1. AlexNet model

AlexNet introduced the ReLU activation function and dropout, revolutionizing deep learning. It has a specific layer configuration with five convolutional layers with 3x3 max pooling layers each and three fully connected layers. AlexNet model utilizes large-size filters (11x11 and 5x5) and then 3x3 filters for the last three convolutional layers. Despite its success, AlexNet's drawbacks include many hyper-parameters (up to 60 million) and relatively shallow depth, resulting in challenges for complex feature learning and slower convergence compared to newer models [13][16].

2. Visual Geometry Group (Vgg) Network architecture

The VggNet addresses the issue of excessive hyperparameters in previous models by using smaller filters in its convolutional layers (2x2 or 3x3), along with the ReLU function, to enhance non-linearity and improve CNN performance. Each convolutional layer is followed by a max pooling layer, and the network ends with two fc layers of 4096 neurons each, and the last layer of 1000 neurons (ImageNet classification). VGG-n refers to different variants of the VggNet model with varying numbers of layers. The use of small filters reduced computational complexity, but the deeper architecture resulted in high computational demands and training time. VggNet also faced challenges like the vanishing gradient problem [18][21].

3. Residual Network model

The ResNet architecture was designed to handle ultra-deep architectures with many layers, ranging from 16 to 2152 layers. ResNet50 and ResNet101 are popular ResNet models with 50 and 101 layers respectively. The ResNet model pioneered the use of skip connections and residual training, which address the vanishing gradient problem commonly encountered in deep neural networks. Skip connections bypass certain layers, allowing information to flow directly from earlier to later layers. ResNet is built by stacking residual blocks, which leverage skip connections to learn residual mappings. This approach enables faster training and improved accuracy by focusing on new features instead of relearning previously learned ones. ResNet's skip connections revolutionized deep CNN design, allowing for more layers without the vanishing gradient problem. It achieved state-of-the-art performance and won the ILSVRC-2015 competition for computer vision tasks [16][17][19].

4. GoogleNet

GoogleNet introduced the concept of inception modules, which allow for the efficient use of computational resources. It is also known as the Inception model. It changed the way convolutional layers are stacked such that parallel groups of convolutional, pooling, and merge operations were used. This reduces the number of parameters to 4 million, which is 12 times less than AlexNet [19]. InceptionV1 has 22 layers with 1x1, 3x3, and 5x5 filters in parallel, max-pooling layers with padding, and batch normalization. In the InceptionV2 model, two 3×3 convolutions replaced the 5x5 convolution filter to reduce computation time thus increasing speed. Also, batch normalization (BN) was added for stability. In the InceptionV3 model, factorized convolution was introduced where n×n convolutions are factorized into 1×n and n×1 convolutions. This reduces the number of parameters and computational costs. Inception-v4 model introduced reduction blocks to simplify the Inception-V3 model. It used uniform layers and reduced the number of blocks. The default input size for Inception-v4 is 299x299 [20][21].

5. MobileNet

MobileNet model is a lightweight CNN model aimed at efficient deployment on mobile devices with limited computational resources [10]. Open-sourced by Google, it uses depthwise separable convolution (DSC) to reduce parameters and create a lightweight deep neural network. The model uses batch normalization and ReLU nonlinearity on all layers except the final fully connected layer, which feeds into a SoftMax layer for classification. MobileNet has a default input size of 224x224. The model has been improved with new features, such as hard-swish activation function, squeeze-and-excitation block, and dynamic convolution, and has achieved state-of-the-art performance on computer vision tasks like image classification, object detection, and semantic segmentation. Overall, MobileNet models are a powerful and efficient tool for deep learning applications, particularly in resource-constrained environments, outperforming GoogleNet and VggNet. [10][23].

6. EfficientNet

EfficientNet models are highly efficient implementations of convolutional neural networks (CNNs) with large layers ranging from 237 to 813 layers for EffficientNet-B0 to EfficientNet-B7. These models use MBConv, an inverted residual bottleneck block with depth-wise separable convolution and employ a scaling technique to evenly scale every dimension of the network. The compound coefficient θ ensures uniform scaling of width, depth, and resolution. EfficientNets achieve remarkable accuracy, with an 84.3% accuracy over ImageNet, and smaller memory requirements. The model has gained significant attention and validation [14][22].

C. Transfer Learning for Waste Classification:

Transfer learning approach enables models to adjust to new datasets by utilizing knowledge acquired from previously trained models [5]. Pre-trained CNN models, trained on millions of images, can be fine-tuned to classify new images [5][10]. One popular approach is the integration of pre-trained models with additional layers to fine-tune the network for specific waste classification tasks [19]. Even with small datasets, transfer learning allows the CNN models to learn generic features from the source task and adapt them to the specific waste classification task, resulting in improved accuracy and efficiency [17][19]

IV. DATA AND EXPERIMENT SETUP

The objective of this research paper is to classify solid waste that is typically produced in residential areas, aiming to maximize the recycling of valuable natural resources. To provide a comprehensive comparison, we conducted experiments using eight different CNN models, each repeated three times at different numbers of epochs (5, 7, and 10 respectively), resulting in a total of twenty-four experiments. MATLAB was used as the programming environment for carrying out these experiments. The pre-trained models were fine-tuned to fit the classes presented by our dataset. Pretrained models are trained through transfer learning approach.

TrashNet dataset [6] was used. It has images of solid waste in six categories: glass, paper, cardboard, plastic, metal, and trash. Before training the CNN models, the TrashNet dataset underwent preprocessing and augmentation. This involved resizing all images to a size that matches the input of each model. Additionally, data augmentation techniques (see Table1) were applied to increase the diversity of the dataset and mitigate overfitting. The TrashNet dataset was randomly divided into 75% and 25% for training and validation sets respectively. The training set was used to adjust the weights of the neurons in the CNN model, while the validation set was used to assess the model's performance. Table 1 shows the setting for data augmentation and training parameters for the models.

Table 1: Data Augmentation and Training Parameters

Data Augmentation	
Rotation Range	[-45 45]
Scale Range	[1 2]
Random X Reflection	True
Random Y Reflection	True
Random X Scale	[1 1]
Random Y Scale	[1 1]
Image Resize	model's input size

Training Parameters	
Optimizer	sgdm
Learning Rate	0.001
Number of Epoch	5, 7, 10
Minibatch Size	20
Validation Frequency	50
Output Network	best validation loss
Shuffle	every epoch

V. RESULT AND DISCUSSION

This section presents the experiments done. The accuracy and loss values were used to evaluate the performance of each model. A confusion matrix was used visual aid on how well the models predict the classes of the images. Table 2 shows the results of the experiment. The training and validation accuracies of each model are presented.

Analyzing Table 2, ResNet50 achieved the highest validation accuracy of 90.95% after 5 epochs (470 iterations). This was closely followed by ResNet101 with an accuracy of 90.48%. The remaining six CNN architectures exhibited accuracies ranging from 64% to 89%.

After 7 epochs having 658 iteration steps, ResNet50 again achieved the highest validation accuracy of 92.06%. ResNet101 and InceptionV3 both achieved an accuracy of 91.59%, while EfficientNetB0 improved to 90.63%. Most models showed a slight increase in accuracy, except for AlexNet, which experienced a decrease of approximately 2%.

Training models for 10 epochs having 940 iterations, ResNet101 achieved the highest validation accuracy of

92.38%, followed closely by ResNet50 at 92.02%. EfficientNetB0 and InceptionV3 achieved an equal validation accuracy of 91.43%. The other models also demonstrated improved validation accuracy, as shown in Table 2. It can be observed that the models when trained for 10 epochs achieved the highest validation accuracies.

ResNet50 achieved the same high accuracy of 92.06% when it was trained for 7 epochs and 10 epochs respectively. This indicates that the model's performance stabilized after the 7 epochs, as further training did not lead to significant improvements in accuracy. The consistent accuracy suggests that the model reached its optimal performance level and was able to maintain it even with additional training. MobileNetV2 achieved a validation accuracy of 88.25% for both training at 5 epochs and 10 epochs. Despite this improvement, MobileNetV2 achieved good accuracy when compared to all the other models that were trained. This suggests that while MobileNetV2 may not have achieved the highest accuracy among the models, it still demonstrated a consistent and competitive performance overall.

Figure 3 and Figure 4 show the training accuracy and validation accuracy respectively when the CNN models were trained at epoch of 5, 7, and 10. Figure 5 to Figure 7 shows the training loss and validation loss for the CNN models when trained for 5 epochs, 7 epochs, and 10 epochs respectively.

The confusion matrix provides a visual representation of the model's performance on the validation dataset, enabling us to analyze the predicted outcomes concerning the actual labels. Figure 8 to Figure 10 shows the confusion matrix for the CNN models that produced the highest validation accuracy when trained at 5, 7, and 10 epochs respectively.

Table 1: Experiment results

CNN model	Training Accuracy			Validation Accuracy		
	5 Epoch	7 Epoch	10 Epoch	5 Epoch	7 Epoch	10 Epoch
AlexNet	69.1101	74.1298	79.3070	64.2857	62.6825	73.4921
GoogleNet	79.4713	94.8356	79.6041	88.5147	88.8889	90.3175
ResNet18	94.9026	89.7766	99.9667	85.8730	88.5714	89.8413
MobileNetV2	84.5836	79.5606	94.8608	88.2540	88.2540	90.9524
ResNet50	94.9281	99.9914	94.9458	90.9524	92.0638	92.0635
ResNet101	99.9539	89.7980	99.9515	90.4762	91.5873	92.3810
EfficientNetB0	69.1190	84.4680	99.9105	89.5238	90.6349	91.4286
InceptionV3	84.7911	94.8583	94.7275	89.2063	91.5873	91.4286

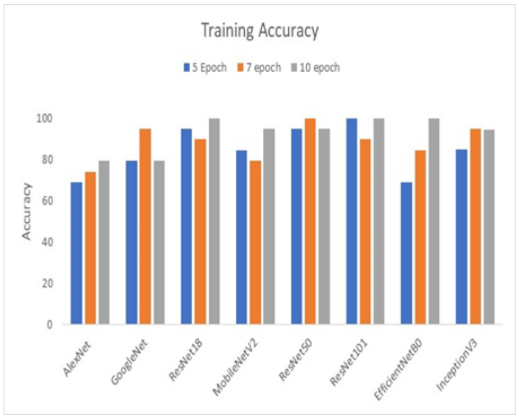

Figure 3: Training Accuracy for Epoch = 5, 7, 10

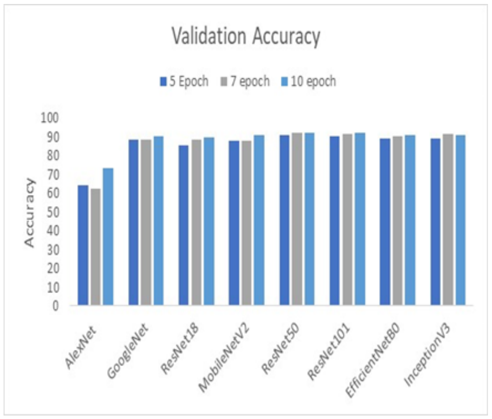

Figure 4: Training Accuracy for Epoch = 5, 7, 10

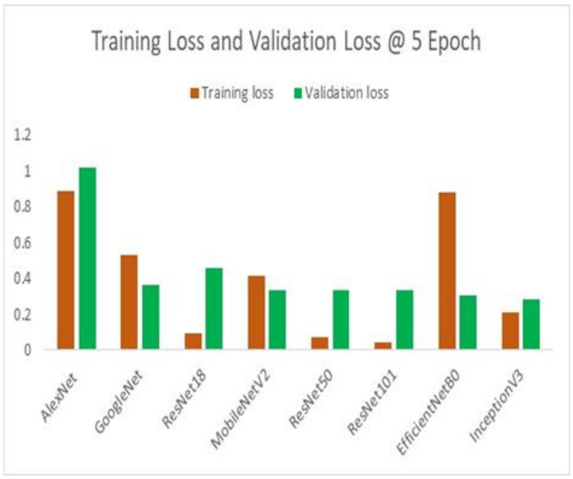

Figure 5: Training and Validation loss for epoch =5

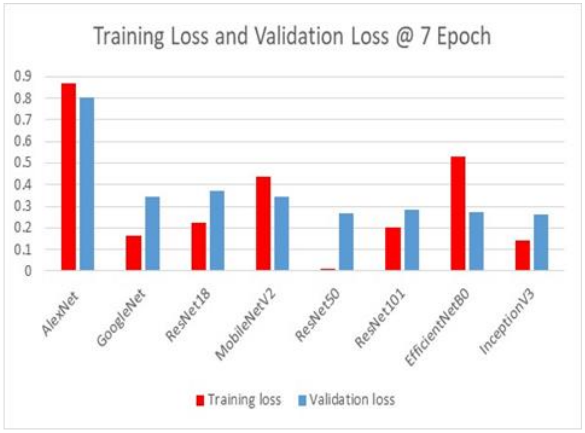

Figure 6: Training and Validation loss for epoch =7

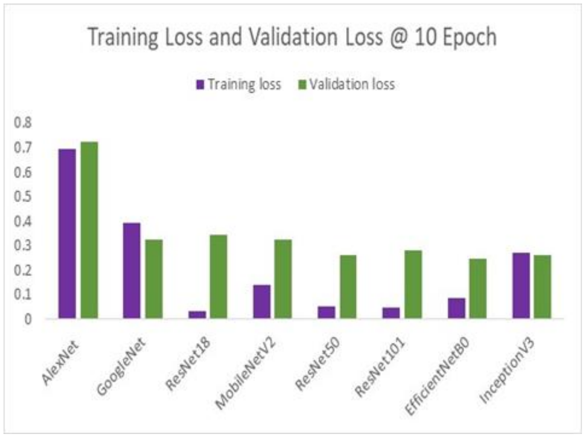

Figure 7: Training and Validation loss for epoch =10

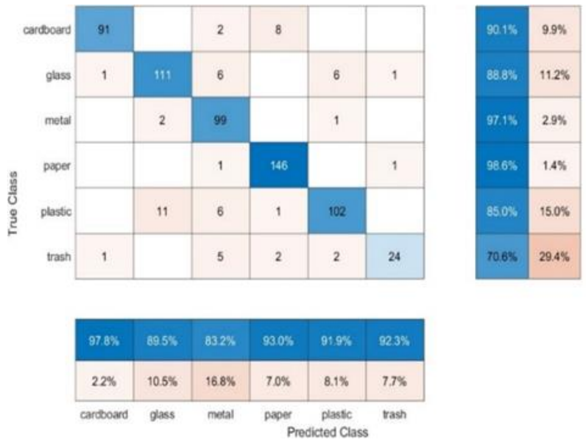

Figure 8: Confusion Matrix for validation data when epoch=5; highest accuracy= 90.95% (ResNet50)

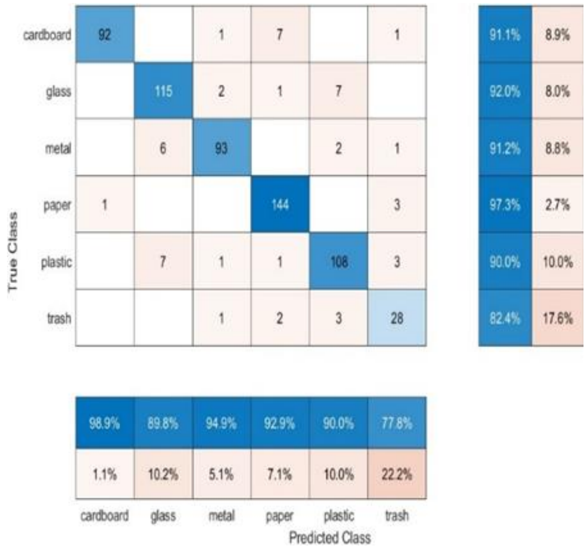

Figure 9: Confusion Matrix for validation data when epoch=7; highest accuracy= 92.06% (ResNet50)

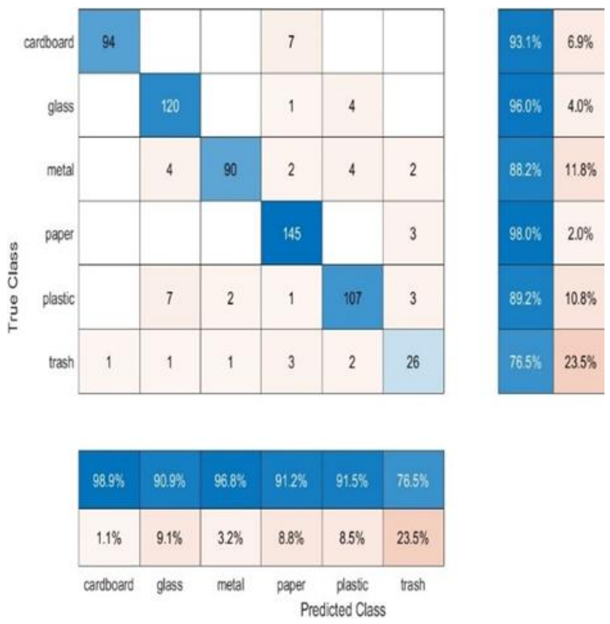

Figure 10: Confusion Matrix for validation data when epoch=10; highest accuracy= 92.38% (ResNet101)

A. *Real world applications*

The CNN models are useful for real-world applications for solid waste classification in waste management processes. Some areas of applications include the following:

i. Waste Management Industry: CNN models for waste classification can be applied in the waste management industry to automate the sorting of waste in recycling plants. This can improve the efficiency of the recycling process and decrease the amount of waste sent to landfills.
ii. Smart Recycling Bins and Robotics: The CNN models can also be integrated with smart bins and robotics to identify and sort waste as it is being thrown away. This eliminates the need for manual sorting, reducing the time and labor required. It can lead to a more efficient and cost-effective waste management system. This can encourage people to recycle correctly and reduce the amount of contaminated waste.
iii. Smart Waste Collection System: CNN models can be integrated into smart waste collection systems that can identify and calculate collection routes for efficient waste collection. It can also be integrated to identify landfill capacity.

CONCLUSION

CNN models have shown great potential for solid waste classification. The experiment was carried out in a MATLAB programming environment. Eight CNN models were trained with the TrashNet dataset for 5, 7, and 10 epochs respectively to show the effect of epoch numbers on the accuracy of the model. TrashNet proved to be an effective dataset for training CNN models for waste classification. In the experiments, ResNet50 achieved better validation accuracy of 90.95% and 92.06% after training for 5 and 7 epochs respectively. ResNet101 achieved highest validation accuracy of 92.38% after training for 10 epochs. The skip connections of the ResNet models contribute to give good accuracy. MobileNetV2 showed an average accuracy when compared with the other model. However, all the models show potential for real world applications such as into smart bins, waste collection processes, and other waste management processes. With continuous advancement in deep learning techniques, CNN models are still powerful tools for solid waste classification and have the potential to revolutionize waste management and aid recycling efforts.

REFERENCES

[1] Ike, C. C., Ezeibe, C. C., Anijiofor, S. C., & Daud, N. N. N. (2018). Solid Waste Management in Nigeria: Problems, Prospects, and Policies. *The Journal of Solid Waste Technology and Management*, *44*(2), 163–172.

[2] Saleh, H. M., & Hassan, A. I. (2021). Introduction Chapter: Solid Waste. In H. M. Saleh (Ed.), *Strategies of Sustainable Solid Waste Management* (pp. 1 – 4). IntechOpen.

[3] Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., Santamaría, J., Fadhel, M. A., Al-Amidie, M., & Farhan, L. (2021). Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. *Journal of Big Data*, *8*(1), 1–74.

[4] Yang, X. S. (2019). Neural networks and deep learning. *Introduction to Algorithms for Data Mining and Machine Learning*, 139–161.

[5] Stancic, A., Vyroubal, V., & Slijepcevic, V. (2022). Classification Efficiency of Pre-Trained Deep CNN Models on Camera Trap Images. *Jounal of Imaging, 8*, 20.

[6] Yang, M., & Thung, G. (2016). *Classification of trash for recyclability status* (CS229 project report). Stanford University. http://cs229.stanford.edu/proj2016/report/ThungYang-ClassificationOfTrashForRecyclabilityStatus-report.pdf

[7] Rahman, M. W., Islam, R., Hasan, A., Bithi, N. I., Hasan, M. M., & Rahman, M. M. (2020). Intelligent waste management system using deep learning with IoT. *Journal of King Saud University - Computer and Information Sciences*, 1–16.

[8] Gyawali, D., Regmi, A., Shakya, A., Gautam, A., & Shrestha, S. (2020). *Comparative analysis of multiple deep CNN models for waste classification*. ArXiv preprint.

[9] Bhandari, S. (2020). Automatic Waste Sorting in Industrial Environments Via Machine Learning Approaches, [master's thesis, Tampere University]

[10] Wang, C., Qin, J., Qu, C., Ran, X., Liu, C., & Chen, B. (2021). A smart municipal waste management system based on deep-learning and Internet of Things. *Waste Management*, *135*, 20-29.

[11] Radhika, D. A. (2021). Real Life Smart Waste Management System [Dry, Wet, Recycle, Electronic & Medical]. *International Journal of Scientific Research in Science and Technology*, *7*(4), 631–640.

[12] Geetha, S., Saha, J., Dasgupta, I., Bera, R., Lawal, I. A., & Kadry, S. (2022). Design of Waste Management System Using Ensemble Neural Networks. *Designs*, *6*(2), 27.

[13] Ravichandiran, S. (Ed.). (2019). Introduction to Deep Learning. In *Hands-On Deep Learning Algorithms with Python* (pp. 7–39). Van Haren Publishing.

[14] Abiodun, O. I., Jantan, A., Omolara, A. E., Dada, K. V., Mohamed, N. A., & Arshad, H. (2018). State-of-the-art in artificial neural network applications: A survey. *Heliyon*, *4*(11), e00938.

[15] Islam, M., Chen, G., & Jin, S. (2019). An Overview of Neural Network. *American Journal of Neural Networks and Applications*, *5*(1), 7–11.

[16] Yamashita, R., Nishio, M., Do, R. K. G., & Togashi, K. (2018). Convolutional neural networks: an overview and application in radiology. *Insights into Imaging*, *9*(4), 611–629.

[17] Gavali, P., & Banu, J. S. (2019). Deep Convolutional Neural Network for Image Classification on CUDA Platform. *Elsevier eBooks*, 99-122.

[18] Xia, W., Jiang, Y., Chen, X., & Zhao, R. (2022). Application of machine learning algorithms in municipal solid waste management: A mini review. *Waste Management & Research, 40*(6), 609 – 624

[19] Chen, L., Li, S., Bai, Q., Yang, J., Jiang, S., & Miao, Y. (2021). Review of Image Classification Algorithms Based on Convolutional Neural Networks. *Remote Sensing*, *13*(22), 4712.

[20] Zheng, X., & Cloutier, R. S. (2022). A review of image classification algorithms in IoT. *EAI Endorsed Transactions on Internet of Things, 7*(28), 1-11

[21] Dhillon, A., & Verma, G. K. (2019). Convolutional neural network: a review of models, methodologies, and applications to object detection. *Progress in Artificial Intelligence*, *9*(2), 85–112.

[22] Imane, C. (2022, November 12). *EfficientNet [model architecture]*. OpenGenus IQ: Computing Expertise & Legacy. Retrieved January 20, 2023, from https://iq.opengenus.org/efficientnet/.

[23] Phiphiphatphaisit, S., & Surinta, O. (2020). Food Image Classification with Improved MobileNet Architecture and Data Augmentation. *Proceedings of the 2020 3rd International Conference on Information Science and Systems* (pp. 51-56).