# Comparative Analysis of Transfer Learning Hyperparameters for Image Classification

Neelesh Karthikeyan
Department of Industrial Engineering
College of Engineering Guindy
Chennai, India
neelesh.karthi@gmail.com

***Abstract*—With the rapid rise in urbanization, garbage generation has increased exceedingly. This study aims to develop a Convolutional Neural Network (CNN) for classifying images of trash into biodegradable and non-biodegradable wastes. Transfer Learning methods based on CNN have shown promising outcomes on diverse image classification problems. This paper reviews and compares the performance of twenty transfer learning models in the Keras library. The results show that the model based on NASNetMobile had the highest accuracy of 96%, while models based on Residual Networks had the least accuracy of 63%. Further, the model's learning rate, batch size, activation function, and optimizer were tuned. The two-factor ANOVA shows that the main effects of learning rate are significant. In contrast, the main effects of batch size and their interaction effect are not significant in a model's accuracy.**

***Keywords—Machine Learning, Trash Classification, Analysis of Variance, Hypothesis Testing, CNN, Hyper-parameter, Transfer Learning***

## I. INTRODUCTION

Waste segregation has become an increasingly widespread problem in recent years. According to a study published by the World Bank in 2018 [1], 242 million tonnes of plastic waste were produced globally in 2016, accounting for 12% of total solid waste. Biodegradable wastes are decomposed by natural forces such as fire, water, air, microorganisms, and soil within a few months. While non-biodegradable wastes take more than 100 years to decompose depending on the type of material. Convolutional Neural Network (CNN) models are used in deep learning to train and test large datasets of images [2]. Every input image will be processed by a set of convolutional layers that include filters, pooling layers, and fully connected layers [3]. Transfer learning is a powerful deep learning technique that can aid in solving the problem of insufficient training data. Because of its wide range of applications, transfer learning has become a promising area in machine learning [4]. This paper aims to review the Transfer learning models in the Keras library and train CNNs to sort waste into biodegradable and non-biodegradable categories. In addition, the impact of hyper-parameters like activation function, batch size, learning rate, and optimizers on waste image classification has been investigated. The effect of learning rate and batch size on a model's accuracy is studied using a two-factor analysis of variance. The best hyper-parameters are selected based on the CNN model's train, test, and validation accuracy. The following is a breakdown of the paper's structure. The second section discusses a review of the related works. In the third section, the dataset and image classification methodologies are demonstrated. The concluding section summarizes the findings of this study..

## II. RELATED WORKS

TrashNet is a dataset created by Yang and Thung [5] containing 2527 images divided into six categories: glass, paper, cardboard, plastic, metal, and trash. The TrashNet dataset has been explained in detail in the upcoming sections.

**Table 1.** Comparison of different network architectures

| S. No | Model | Year | Parameters |
|---|---|---|---|
| 1 | LeNet-5 | 1998 | 60,850 |
| 2 | AlexNet | 2012 | 62,378,344 |
| 3 | VGG16 | 2014 | 138,357,544 |
| 4 | VGG19 | 2014 | 143,667,240 |
| 5 | ResNet50 | 2015 | 25,636,712 |
| 6 | ResNet101 | 2015 | 44,707,176 |
| 7 | ResNet152 | 2015 | 60,419,944 |
| 8 | Xception | 2016 | 22,910,480 |
| 9 | ResNet50V2 | 2016 | 25,613,800 |
| 10 | ResNet101V2 | 2016 | 44,675,560 |
| 11 | ResNet152V2 | 2016 | 60,380,648 |
| 12 | InceptionV3 | 2016 | 23,851,784 |
| 13 | DenseNet121 | 2017 | 8,062,504 |
| 14 | DenseNet169 | 2017 | 14,307,880 |
| 15 | DenseNet201 | 2017 | 20,242,984 |
| 16 | InceptionResNetV2 | 2017 | 55,873,736 |
| 17 | MobileNet | 2018 | 4,253,864 |
| 18 | MobileNetV2 | 2018 | 3,538,984 |
| 19 | NASNetMobile | 2018 | 5,326,716 |
| 20 | NASNetLarge | 2018 | 88,949,818 |

Bircanoglu et al. [6] developed RecycleNet, a lightweight CNN model for trash classification. Even though RecycleNet only achieved around eighty percent accuracy on the TrashNet, it reduced time complexity by reducing the number of parameters from seven to seven to three million. AlexNet was proposed in 2012 by Alex Krizhevsky et al., and it performed well in the image categorization challenge [7]. Kennedy et al. used the Visual Geometry Group 19 (VGG-19) [8] to classify trash with 88% accuracy [9]. The significance of batch size on image classification was reviewed by Radiuk [10] in 2017. According to the author's findings, the larger the batch size, the higher the network accuracy. A batch size of thirty-two, according to Bengio [11], is a good starting point, and the number can be tuned further. The LeNet-5 neural network was proposed by Yann LeCun, Leon Bottou, Yosuha Bengio, and Patrick Haffner in 1998 [12]. It was one of the earliest neural networks, containing sixty thousand parameters. The seven layers include three convolutional layers, two pooling layers, and one fully connected layer. There are eighty-four units in the fully connected layer [13]. The AlexNet was founded in 2012 and had over sixty-two million parameters. This network achieved a top-5 error of 15.3% in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [14]. The ILSVRC was used to develop this model, which used a part of the ImageNet database. The data holds 1.4 million images over one thousand object classes. The ReLU (Rectified Linear Unit) function [14] introduces non-linearity in the network. VGG16 and VGG19 networks were proposed by K. Simonyan et al. and A. Zisserman et al. [15]. The weight layers in VGG16 are thirteen convolutional layers, three fully connected layers, and a softmax layer. VGG19, on the other hand, has nineteen layers: sixteen convolutional layers, three fully connected layers, and a softmax layer. There are over 138 million parameters in VGG16 and 143 million in VGG19. There are sixty-four filters of size $3 \times 3$ in the first two layers. The input image dimension is changed to $224 \times 224 \times 64$ with a max-pooling layer of stride two. The residual Neural Network (ResNet) network introduces 'residual learning' [16]. The fifty-layer ResNet won the ILSVRC in 2015. There are four stages to the ResNet50 model, including a Convolution and Identity block. Each convolution block and each identity block have three convolution layers. In the ResNet architecture, the first convolution is done with $7 \times 7$ size filters. At the same time, a $3 \times 3$ size filter is used to perform max pooling. To improve the network's performance, the Deep Residual Network uses bottleneck residual block architecture. InceptionV3, also known as GoogleNet, is an upgraded version of InceptionV1 [17]. The InceptionV3 model consists of a basic convolutional block, an improved Inception module, and a classifier. There are forty-eight layers and over twenty-four million parameters in it. Inception is divided into three modules: Inception-A, Inception-B, and Inception-C. The eleven convolutional kernels are commonly used to reduce the number of feature channels and speed up training [18]. At the end of the Inception modules, three fully connected layers allow to use of the pre-trained model and fine-tune the parameters [19]. InceptionResNetV2, also known as the hybrid inception model, is designed on the Inception architecture while additionally including residual connections. Each branch has a concatenated set of filters ranging from $1\times1$ to $3\times3$, $5\times5$. This model includes Inception ResNet-A, Inception ResNet-B, and Inception ResNet-C [19]. Each Inception block is linked to a $1\times1$ filter expansion layer, which is used to scale up the dimensions of the filter bank before adding it. Xception includes a redesigned inception block that replaces several spatial dimensions such as $1\times1$, $5\times5$, and $3\times3$ with a single dimension followed by a 1x1 convolution to reduce computational complexity. Entry Flow, Middle Flow, and Exit Flow are the three major blocks. CNN's are used in thirty-six layers. Except for the first and last modules, these layers are divided into 14 modules surrounded by linear residual connections. The data first passes through the entering flow, then through the middle flow eight times, and finally through the exit flow. All convolution and depth-wise separable Convolution layers are batch normalized [20]. With 132 layers and over eight million parameters, DenseNet121 is a densely connected convolutional network [21]. In the DenseNet design, the first is the convolution block, which is the same as the identity block in ResNet. The dense block, in which the convolution blocks are concatenated and densely coupled, is the second part. The Dense block is the principal component of the DenseNet architecture [21]. The final layer is the transition layer, connecting two continuous dense blocks. The use of a transition block reduces the dimension of feature maps. With over three million parameters [23], MobileNetV2 is based on a linear bottleneck layer and an inverted residual structure [22]. There are two types of blocks in MobileNetV2. The first, with a stride of one, is the residual block. A residual block with a stride of two for shrinking and three layers in each block is another choice. A ReLU 1x1 convolutional layer [22] is the first layer. The depth-wise convolution layer employs $3\times3$ depth-wise separable convolutions, and the third layer is a linear $1\times1$ convolutional layer. NasNetMobile is a scalable CNN consisting of twelve blocks and over five million parameters. Depending on the network's capacity requirements, each block consists of a few basic processes repeated numerous times. The block is the smallest unit in NASNet [23][24]. NasNetLarge is a CNN with over eighty-eight million parameters trained on over a million pictures from the ImageNet database. Normal cell and reduction cell are two of the themes [25]. Normal cells return a feature map in the same dimension as convolutional cells. In reduction cells, convolutional cells return a feature map with dimensions lowered by two. The NAS-Neural Network Search algorithm [26] identifies the best neural network architecture. The RNN controller of the NAS algorithm samples the blocks and assembles them to construct an end-to-end architecture. The list of pre-trained CNN architectures available in the Keras library, with the year of publication and number of parameters, are summarized in Table 1.

## III. Methodology

The images in this dataset were taken with a variety of mobile devices. The photos were taken in front of a white background with natural or artificial lighting. The original dataset is over 3.5 GB in size, and each image has been reduced to $512 \times 384$ pixels in size. The data distribution is shown in Table 2, while Figure 1 shows examples from each class in this dataset.

**Table 2.** Distribution of data in TrashNet

| S. No | Class Name | Number of Images |
|---|---|---|
| 1 | Cardboard | 403 |
| 2 | Paper | 594 |
| 3 | Glass | 501 |
| 4 | Plastic | 482 |
| 5 | Metal | 410 |
| 6 | Trash | 137 |

Data Augmentation artificially increases the size of the training dataset, allowing the model to learn and generalize better on future unseen data [27]. To eliminate any bias in the model, Upsampling was used. It makes sure that both classes had the same number of images before training.


**Figure 1:** Example samples of TrashNet Dataset

Random rotation, brightness adjustment, horizontal & vertical flips, channel shift, horizontal & vertical shifts, and channel shift were among the image manipulation techniques used. While increasing the dataset size, these picture changes were used to accommodate the various material orientations. Biodegradable and non-biodegradable photos were further divided into two categories.

**Figure 2**: Augmented images of a glass sample

Metal, glass, and plastic are non-biodegradable by nature, while paper and cardboard are. The final data- set contains 19500 photos, with 9750 belonging to class one (biodegradable wastes) and the remaining 9750 to class two (non-biodegradable wastes). The dataset was split into three parts: 80% for training, 10% for validation, and 10% for testing. Figure 2 shows the augmented images of one sample from the glass class.

**Table 3.** Validation and Training Accuracy

| S. No | Model | Validation Accuracy | Training Accuracy |
|---|---|---|---|
| 1 | NASNetMobile | 0.9236 | 0.9709 |
| 2 | NASNetLarge | 0.9128 | 0.9628 |
| 3 | DenseNet201 | 0.9108 | 0.9467 |
| 4 | DenseNet121 | 0.8964 | 0.9375 |
| 5 | DenseNet169 | 0.8933 | 0.9287 |
| 6 | ResNet50V2 | 0.8856 | 0.9475 |
| 7 | ResNet101V2 | 0.8851 | 0.9437 |
| 8 | MobileNet | 0.8836 | 0.9308 |
| 9 | ResNet152V2 | 0.8826 | 0.9382 |
| 10 | Xception | 0.8821 | 0.9497 |
| 11 | MobileNetV2 | 0.8682 | 0.9285 |
| 12 | InceptionResNetV2 | 0.8554 | 0.8998 |
| 13 | InceptionV3 | 0.8549 | 0.8815 |
| 14 | LeNet-5 | 0.8174 | 0.8712 |
| 15 | VGG19 | 0.8144 | 0.8675 |
| 16 | VGG16 | 0.8108 | 0.8628 |
| 17 | AlexNet | 0.7354 | 0.8061 |
| 18 | ResNet50 | 0.6621 | 0.6905 |
| 19 | ResNet101 | 0.5923 | 0.6616 |
| 20 | ResNet152 | 0.5903 | 0.6613 |

**Table 4.** Validation and Training loss

| S. No | Model | Validation Loss | Training Loss |
|---|---|---|---|
| 1 | NASNetMobile | 0.2023 | 0.0780 |
| 2 | NASNetLarge | 0.6230 | 0.1746 |
| 3 | DenseNet201 | 0.2116 | 0.1351 |
| 4 | DenseNet121 | 0.2594 | 0.1535 |
| 5 | DenseNet169 | 0.2555 | 0.1714 |
| 6 | ResNet50V2 | 0.3048 | 0.1349 |
| 7 | ResNet101V2 | 0.2788 | 0.1424 |
| 8 | MobileNet | 0.2829 | 0.1744 |
| 9 | ResNet152V2 | 0.3115 | 0.1526 |
| 10 | Xception | 0.2833 | 0.1310 |
| 11 | MobileNetV2 | 0.3243 | 0.1756 |
| 12 | InceptionResNetV2 | 0.3478 | 0.2311 |
| 13 | InceptionV3 | 0.3454 | 0.2723 |
| 14 | LeNet-5 | 0.4338 | 0.2961 |
| 15 | VGG19 | 0.4282 | 0.2983 |
| 16 | VGG16 | 0.4088 | 0.3055 |
| 17 | AlexNet | 0.5730 | 0.4129 |
| 18 | ResNet50 | 0.6165 | 0.5931 |
| 19 | ResNet101 | 0.6607 | 0.6139 |
| 20 | ResNet152 | 0.6744 | 0.6195 |

Figure 3 shows the variation of accuracy and loss in the training and validation data for over twenty epochs of the TrashNet. At the end of twenty epochs, the model had a training accuracy of 0.985 and a validation accuracy of 0.94.

The loss on the training data is 0.03, while the loss on the validation data is 0.22. This shows over-fitting, which can be solved by adding more images to the data set or adding more features that retain the image's original dimensions. The twenty models were trained for up to five epochs, and the accuracies and losses are tabulated in Table 3. NASNetMobile was the best performing model with the highest validation accuracy and least validation loss. ResNet152 had the least accuracy among the twenty models.

## IV. Results

**Table 5.** Test data accuracy and precision

| S. No | Model | Test Accuracy | Precision |
|---|---|---|---|
| 1 | NASNetMobile | 0.96205 | 0.95004 |
| 2 | NASNetLarge | 0.96154 | 0.94379 |
| 3 | DenseNet201 | 0.95333 | 0.94467 |
| 4 | DenseNet169 | 0.95333 | 0.94378 |
| 5 | Xception | 0.93180 | 0.92440 |
| 6 | DenseNet121 | 0.93128 | 0.91675 |
| 7 | ResNet101V2 | 0.92923 | 0.90671 |
| 8 | MobileNet | 0.92718 | 0.91859 |
| 9 | ResNet50V2 | 0.92462 | 0.88692 |
| 10 | ResNet152V2 | 0.91333 | 0.88454 |
| 11 | MobileNetV2 | 0.91282 | 0.89345 |
| 12 | InceptionV3 | 0.89641 | 0.89723 |
| 13 | InceptionResNetV2 | 0.89590 | 0.85874 |
| 14 | VGG16 | 0.87949 | 0.89278 |
| 15 | VGG19 | 0.87180 | 0.89189 |
| 16 | LeNet-5 | 0.82923 | 0.79833 |
| 17 | ResNet50 | 0.72256 | 0.72939 |
| 18 | AlexNet | 0.70821 | 0.64732 |
| 19 | ResNet152 | 0.65231 | 0.65138 |
| 20 | ResNet101 | 0.63692 | 0.58824 |

**Table 6.** Test data recall and F1-score

| S. No | Model | Recall | F1-score |
|---|---|---|---|
| 1 | NASNetMobile | 0.97538 | 0.96254 |
| 2 | NASNetLarge | 0.98154 | 0.96229 |
| 3 | DenseNet201 | 0.96308 | 0.95378 |
| 4 | DenseNet169 | 0.96410 | 0.95383 |
| 5 | Xception | 0.94051 | 0.93238 |
| 6 | DenseNet121 | 0.94872 | 0.93246 |
| 7 | ResNet101V2 | 0.95692 | 0.93114 |
| 8 | MobileNet | 0.93744 | 0.92792 |
| 9 | ResNet50V2 | 0.97333 | 0.92812 |
| 10 | ResNet152V2 | 0.95077 | 0.91646 |
| 11 | MobileNetV2 | 0.93744 | 0.91492 |
| 12 | InceptionV3 | 0.89539 | 0.89630 |
| 13 | InceptionResNetV2 | 0.94769 | 0.90102 |
| 14 | VGG16 | 0.86256 | 0.87741 |
| 15 | VGG19 | 0.84615 | 0.86842 |
| 16 | LeNet-5 | 0.88103 | 0.83764 |
| 17 | ResNet50 | 0.70769 | 0.71838 |
| 18 | AlexNet | 0.91487 | 0.75818 |
| 19 | ResNet152 | 0.65538 | 0.65337 |
| 20 | ResNet101 | 0.91282 | 0.71543 |

The trained models were tested on the test data, and the predictions were compared with the true labels. The accuracy metrics for each model are tabulated in Table 4. The NASNetMobile was the best-performing model.

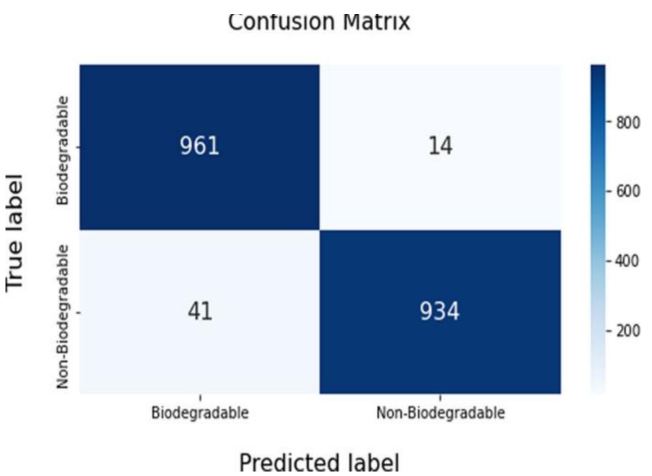


**Figure 3.** Confusion matrix with true vs predicted label for NASNetMobile

The confusion matrix, shown in Figure 5, is a summary of prediction results on a classification problem. The model was tested using different activation functions on a smaller dataset and the accuracy metrics are tabulated in Table 5. Among the four activation functions, the model using Exponential Linear Unit (ELU) activation function had the highest accuracy. Table 6 describes the accuracy metrics of the model when different optimizers were used. The Adam optimizer, out-performed RMSprop and SGD. Three different learning rates were used while training the model and the accuracy metrics are shown in Table 7. From Table 8, it can be found that the model had the highest accuracy when the batch size was 32.

**Table 7:** Comparisons against activation functions

| Network | Validation Accuracy | Training Accuracy | Test Accuracy |
|---|---|---|---|
| **ELU** | 0.96500 | 0.96071 | 0.95 |
| **ReLU** | 0.96250 | 0.95643 | 0.95 |
| **LeakyReLU** | 0.96250 | 0.95214 | 0.9475 |
| **SELU** | 0.95750 | 0.94357 | 0.9425 |

**Table 8:** Comparisons against different optimizers

| Optimizer | Validation Accuracy | Training Accuracy | Test Accuracy |
|---|---|---|---|
| **Adam** | 0.970 | 0.958571 | 0.943077 |
| **RMSprop** | 0.965 | 0.943571 | 0.938974 |
| **SGD** | 0.960 | 0.955714 | 0.9451281 |

**Table 9:** Comparisons against different learning rates

| Learning Rate | Validation Accuracy | Training Accuracy | Test Accuracy |
|---|---|---|---|
| **0.001** | 0.9625 | 0.957857 | 0.9475 |
| **0.0001** | 0.945 | 0.965 | 0.9375 |
| **0.01** | 0.94 | 0.952857 | 0.9175 |

**Table 10:** Comparisons against different batch sizes

| Batch Size | Validation Accuracy | Training Accuracy | Test Accuracy |
|---|---|---|---|
| **32** | 0.9725 | 0.963571 | 0.9375 |
| **64** | 0.9675 | 0.963571 | 0.955 |
| **128** | 0.9625 | 0.962143 | 0.955 |

A two-factor factorial experiment was conducted to study the effect of batch size and learning rate on the accuracy of a neural network. The two factors are investigated at three levels each. The following data (Table 9) are obtained from two replications. The effect of a factor is defined as the change in response due to a change in the level of the factor [28]. The combined effect due to both the factors is called the interaction effect.

**Table 11.** Accuracy (%) for different batch size and learning rate combination

| **Accuracy (%)** | | **Batch size** | | |
|---|---|---|---|---|
| | | **32** | **64** | **128** |
| **Learning Rate** | **0.0001** | 96.4103 | 95.3333 | 95.6923 |
| | | 94.7179 | 95.8974 | 95.4872 |
| | **0.001** | 95.8974 | 95.4872 | 95.4359 |
| | | 95.4359 | 95.3846 | 95.641 |
| | **0.01** | 93.6923 | 94.9744 | 95.0769 |
| | | 90.9744 | 94.7179 | 93.8462 |

**Table 12.** ANOVA for Two-factor experiment

| Source of Variation | Sum of Squares | Degrees of Freedom | Mean Squares | $F_0$ |
|---|---|---|---|---|
| Batch Size | 2.14 | 2 | 1.07 | 1.55 |
| Learning Rate | 11.40 | 2 | 5.70 | 8.26 |
| Batch Size x Learning Rate | 5.24 | 4 | 1.31 | 1.89 |
| Interaction Error | 6.23 | 9 | 0.69 | |
| Total | 25.02 | 17 | | |

Since $F_{0.05,2,9} = 4.26$ and $F_{0.05,4,9} = 3.63$, Learning Rate has got significant effect on the accuracy of a neural network. However, Batch Size and the interaction effect show no significance.

## V. Conclusion

Today's computer vision algorithms enable the classification of trash into several categories. A dataset of about 39000 images was developed as the first contribution of this study. NASNetMobile had the highest test accuracy of 96.20%, with the lowest validation loss of 0.20, among all the models when the various hyper-parameters such as optimizer and loss function were kept constant. Other models, such as NASNetLarge, DenseNet201, and DenseNet169 had accuracy levels over 95%. In image classification, hyper-parameters must be adjusted depending on the dataset used to train a model. For large learning rates, there is a strong link between learning rate and batch size. Based on the findings, it can be stated that the learning rate has a greater influence on accuracy than the batch size. Waste classification is a complicated task with a lot of elements to consider. Unlike other image classification problems using simplified images or quantitative data, waste classification needs to consider the object's surroundings. The model must also avoid the most common object recognition mistakes, such as misidentifying multiple independent objects as a single object or vice versa, finding the same thing in multiple classes, and not detecting hidden objects. However, the success rate in real systems can be lower because of the small amount of data and the white background of all the images.

## VI. References


[1] S. Kaza, L. C. Yao, P. Bhada-Tata, and F. V. Woerden, ''What a waste 2.0,'' World Bank Publications, vol. 30317. Washington, DC, USA: The World Bank, 2018. https://openknowledge.worldbank.org/bitstream/handle/10986/30317/211329ov.pdf

[2] T. Zhou, S. Ruan, and S. Canu, "A review: Deep learning for medical image segmentation using multimodality fusion," Array, vol. 3–4, no. August, p. 100004, 2019, https://doi.org/10.1016/j.array.2019.100004

[3] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, "A survey of the recent architectures of deep convolutional neural networks," Artif. Intell. Rev., pp. 1–70, 2020, https://doi.org/10.1007/s10462-020-09825-6

[4] Y. Zhu et al., "Heterogeneous transfer learning for image classification," 2011.

[5] M. Yang and G. Thung, ''Classification of trash for recyclability status,'' Mach. Learn., Stanford, CA, USA, Project Rep. CS229, 2016.

[6] C. Bircanoglu, M. Atay, F. Beser, O. Genc, and M. A. Kizrak, "RecycleNet: Intelligent waste sorting using deep neural networks," in Proc. INISTA, 2018, pp. 1–7, https://doi.org/10.1109/INISTA.2018.8466276

[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. Adv. Neural Inf. Process. Syst. 2012, 25, pp.1097-1105.

[8] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. arXiv 2014, arXiv:1409.1556.

[9] T. Kennedy, OscarNet: Using transfer learning to classify disposable waste. In CS230 Report: Deep Learning; Stanford University: Stanford, CA, USA, 2018.

[10]P. Radiuk, Impact of training set batch size on the performance of convolutional neural networks for diverse datasets, Inf. Technol. Manag. Sci. 20 (2017), https://doi.org/10.1515/itms-2017-0003

[11] Y. Bengio, Practical recommendations for gradient-based training of deep architectures, 2012, arXiv, https://doi.org/10.1007/978-3-642-35289-8_26

[12] Y. Lecun, L. Bottou, Y. Bengio, and P. Ha, "LeNet," Proc. IEEE, no. November, pp. 1–46, 1998, https://doi.org/10.1109/5.726791

[13] A. El-Sawy, H. EL-Bakry, and M. Loey, "CNN for handwritten arabic digits recognition based on lenet-5BT - proceedings of the international conference on advanced intelligent systems and informatics 2016," 2017, pp. 566–575, https://doi.org/10.1007/978-3-319-48308-5_54

[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," Handb. Approx. Algorithms Metaheuristics, pp. 1–1432, 2007, https://doi.org/10.1201/9781420010749

[15] K. Simonyan and A. Zisserman, "Very deep convolutional networks for largescale image recognition," 3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc., pp. 1–14, 2015.

[16] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., vol. 2016-Decem, pp. 770–778, 2016, https://doi.org/10.1109/CVPR.2016.90

[17] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., vol. 2016-Decem, pp. 2818–2826, 2016, https://doi.org/10.1109/CVPR.2016.308

[18] X. Xia, C. Xu, and B. Nan, "Inception-v3 for flower classification," in 2017 2nd International Conference on Image, Vision, and Computing (ICIVC), 2017, pp. 783–787, https://doi.org/10.1109/ICIVC.2017.7984661

[19] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inceptionResNet and the impact of residual connections on learning," 31st AAAI Conf. Artif. Intell. AAAI 2017, pp. 4278–4284, 2017.

[20] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017, vol. 2017-Janua, pp. 1800–1807, 2017, https://doi.org/10.1109/CVPR.2017.195

[21] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017, vol. 2017-Janua, pp. 2261–2269, 2017, https://doi.org/10.1109/CVPR.2017.243

[22] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," Proc. IEEE Comput. Soc. Conf. Comput. Vis.Patter.Recognit.,pp.4510–4520, 2018, https://doi.org/10.1109/CVPR.2018.00474

[23] F. Saxen, P. Werner, S. Handrich, E. Othman, L. Dinges, and A. Al- Hamadi, "Face attribute detection with mobilenetv2 and nasnetmobile," Int. Symp. Image Signal Process. Anal. ISPA, vol. 2019-Septe, no. October, pp. 176–180, 2019, https://doi.org/10.1109/ISPA.2019.8868585

[24] K. Radhika, K. Devika, T. Aswathi, P. Sreevidya, V. Sowmya, and K. P. Soman, Performance analysis of NASNet on unconstrained ear recognition, vol. SCI 871, no. January. Springer International Publishing, 2020, https://doi.org/10.1007/978-3-030-33820-6_3

[25] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., pp. 8697–8710, 2018, https://doi.org/10.1109/CVPR.2018.00907

[26] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," 5th Int. Conf. Learn. Represent. ICLR 2017 - Conf. Track Proc., pp. 1–16, 2019.

[27] M. A. Tanner and W. H. Wong, "The calculation of posterior distributions by data augmentation," J. Am. Stat. Assoc., vol. 82, no. 398, pp. 528–540, Jul. 1987, https://doi.org/10.2307/2289457

[28] K. Krishaiah, P. Shahabudeen, "Applied Design of Experiments and Taguchi Methods", PHI Learning Private Limited, New Delhi, 2012