Journal of Innovative Image Processing (ISSN: 2582-4252)
www.irojournals.com/iroiip/

Image Classification Model Selector

N. Arulanand[1], D. Kamalraj[2], B. Krishna Teja[3]

Computer Science and Engineering, PSG College of Technology, Anna University, Coimbatore, India

E-mail: [1]naa.cse@psgtech.ac.in, [2]kamalrajdhanakodi@gmail.com, [3]b.krishnateja732@gmail.com

Abstract

Image classification is a part of computer vision, in which the digital system categorizes the entire image. Deep Learning (DL) models are widely used for image classification. However, creating DL models is resource-intensive and time-consuming, and requires extensive knowledge in the DL domain. Google Teachable Machines (GTM) is a website that outputs a trained model given the dataset, however, GTM uses only the MobileNet model and does not balance the image dataset which affects the model's accuracy. This paper proposes a tool that automates the steps in building and training an image classification model. Using this tool does not require any extensive knowledge in DL. The tool automates the image data pre-processing steps, model building, model training, and model testing to output the best model for the given image classification dataset based on the test accuracy. The tool is tested on two datasets (each balanced and unbalanced dataset): a custom construction dataset and a Minet dataset. Both datasets are also used to train models using the GTM website. Due to the automated pre-processing steps, the average increase in the accuracy is 14.55% in the construction dataset and 3.91% in the Minet dataset. Comparing to the GTM models, the tool produced model with 8.33% more accuracy on the construction dataset and model with 14.07% more accuracy on the Minet dataset. The models trained by the proposed tool have better accuracy compared to the models obtained using GTM. Thus, using the image classification model selector facilitates the creation of an effective image classification model for the target dataset.

Keywords: Image classification, automated image classification model selection, automated image classification

1. Introduction

The digital system categorizes the entire image in image classification process. Image classification has many applications in the fields of defense, security, medical imaging,

DOI: https://doi.org/10.36548/jiip.2022.4.007
Received: 18.11.2022, received in revised form: 27.12.2022, accepted: 13.01.2023, published: 25.01.2023

automated driving, traffic control systems, and brake-light detection. It plays a vital role in automated tasks. Using image classification in automation helps industries replace some human-centered roles with machines, thereby reducing costs and human errors.

As image classification is the central core of automation works, building Machine Learning (ML) or Deep Learning (DL) models is necessary. To build such models, industries have to hire a ML or DL professional, who can craft the model specifically for the use case. Industries hiring an ML/DL professional for each image classification model building is expensive. Therefore, an automated tool that can construct a deep learning model for the problem at hand can reduce the industrial expenses. Handcrafted deep models created by professionals will be more efficient and accurate than an automatically built model; however, such automated tools reduce the effort. Furthermore, an automated deep learning image classification model building tool can benefit novices in building image classification models. Using such tools, anyone can create an image classification model without any prior knowledge of ML or DL. Such tools can take the image input from the user along with the labels and output a trained model on the given input.

This paper presents the design of an automated image classification model selection tool that takes image input from the user and performs basic processing on the image data, such as image data balancing and image pre-processing, followed by training three pre-trained models (Xception, EfficientNet V2 B3, and MobileNet V3 large) employing transfer learning by changing its output layer. After training these models, the tool provides the model with the best accuracy as the output model.

The remainder of this paper is organized as follows. Section 2 covers the existing solutions and their drawbacks. It also explains the concepts (data imbalances and transfer learning) used to implement the proposed solution and various pretrained deep learning models for image classification, and compares the metrics used to measure the performance of the deep-learning model. Section 3 describes the architecture and implementation of the proposed tool and the datasets used. Section 4 elaborates the results and discussion. Finally, section 5 concludes the paper.

2. Related Works

Google Teachable Machines (GTM) [1] is a web application that helps to build image classification, pose detection, and sound classification models. This web application aims to

help people with little to no knowledge of deep learning to create a deep learning model to classify poses, images, and sounds. This also aims to help people learn the processes in classification model building. The implementation of the GTM uses MobileNet for image classification in the backend to build the model with less resources. To train the model, the GTM uses the concept of transfer learning. Open Mind, AWS machine learning, and Gradient are other tools used to build deep-learning models. Similar to GTM, Open Mind is also a website (hosted on GitHub in url: "https://cluster-11.github.io/open-mind/") that allows users to train models for image classification tasks, but currently it only accepts two input classes. Amazon Machine Learning allows developers to use machine learning. It provides visualization tools and wizards that guide in the process of creating ML models. It makes it easy to obtain predictions using simple APIs. Gradient is a suite of tools for exploring data and training neural networks. The gradient includes one-click Jupyter notebooks, a powerful job runner, and a Python module to run any code on a fully managed GPU cluster in the cloud.

GTM is easy to use, but there are a few drawbacks in using GTM for training a model on the target dataset. The GTM uses the MobileNet model as the base model for transfer learning. The MobileNet models are optimized such that they can be used in embedded devices. However, the other models have better prediction accuracy. If the prediction accuracy is an important metric, then the GTM may not provide the most accurate model for the target dataset. The GTM also does not rectify image data imbalance. This can also affect the performance of the model trained on the target dataset. Although AWS Sagemaker provides extensive tools for training, tuning, and deploying models, it is not beginner friendly. AWS Sagemaker provides GUI tools to build models which still requires basic ML knowledge. Gradient is a platform that makes it easier to build and deploy the models. It does not provide an easy tool to build models without any programming.

2.1 Pre-processing and Data imbalances

In ML or DL, pre-processing steps must be performed on the input data before providing the input data to the models. In the case of image data, resizing the image to the required input size is a basic pre-processing step. Automated deep learning tools are not only required to train models, but also to perform the necessary pre-processing on the input data. In addition to resizing, normalizing and balancing image data are important pre-processing steps for images. Normalization is the process of converting input values into a certain range. In the case of classification, data imbalance indicates nonuniformity in the amount of data for

each class. Some classes have a high number of images, whereas others may be relatively low. In such cases, when the data are used directly, prediction is most favorable for classes with a high data count. For example, medical dataset for disease detection contains more negative cases than positive cases, which favors negative cases more than positive cases on training. To avoid such situations, the data are balanced before being used for training.

Data imbalance can be rectified during the pre-processing [2]. To balance image data, three methods can be used at the data level [3]. The first method is oversampling, in which additional images are added by applying augmentation techniques on images of classes with lower number of images, than the maximum number of images found in any class. The drawback of this method is the addition of duplicate images after the augmentation. The presence of duplicate images can lead to overfitting. The second method is under-sampling, in which the images are randomly removed to obtain equal number of images in all the classes. In under-sampling, images with important features can be lost by the random removal of data. Both oversampling and under-sampling can also be combined, where the images in the class with a higher number of images are reduced by random removal and the images in the class with a lower number of images are increased by augmentation to obtain equal number of images in all the classes. The third method is a different feature selection method for imbalanced datasets. Zheng et al. [4] used this method for text categorization. A different feature selection method was proposed for positive and negative classes and were explicitly combined.

In over-sampling, an effective method used to generate new data is SMOTE, which stands for Synthetic Minority Over-sampling Technique. In SMOTE, new synthetic data was created based on the feature vector of neighboring data. This method proved to be effective when combined with random under-sampling [5]. But using SMOTE will be a costly process than application specific transformation like rotation, changing contrast, as SMOTE may require additional models to create the synthetic data.

2.2 Transfer Learning

Transfer learning uses a model trained on a source domain to improve the model performance in a related target domain. Transfer learning can also be used when there are limitations to the computational resources required to train a model [6]. For image classification, the models trained on the ImageNet dataset (or other domain-specific datasets) are the starting point of the model used for the classification of another target image dataset.

The output layer of these pretrained models is removed, and a new output layer is added based on the number of categories in the target image dataset. The effective model is then trained on the target dataset by either learning all parameters or only the parameters in the newly added output layer. The final model obtained through transfer learning performs well only if the source and target domains are related. The case in which all parameters in the model are learned is termed as fine-tuning. Fine-tuning the model can take more time and requires more computational resources because all parameters are learned during training. Pretrained models for transfer learning are available on the Tensor flow hub platform. The models available in this platform are trained on generic image classification datasets, such as ImageNet, or domain-specific datasets, such as iNaturalist. For generic image classification, models trained on the ImageNet dataset can be used because they cover a wide range of image classes. The ImageNet dataset contains 1000 classes of images and 1,281,167 images. An extended version of the image dataset, Imagenet-21k dataset which contains 21000 classes can also used for pretraining models.

2.3 Deep Learning Models

2.3.1 MobileNet

In deep learning, models based on artificial neural networks contain numerous learnable parameters. Convolution-based network models, which are deep-learning models, are used for image classification. The problem with such models is that they cannot be used in low-resource systems such as mobile phones. Mobile phones have limited memory and computing power due to the form factor. To use deep-learning models in low-resource devices, specialized models with the best trade-off between efficiency and accuracy were built. One such architecture is MobileNet. Using the MobileNet architecture, the models can be directly implemented in applications rather than running the model on a server and performing prediction by API calls. There are three versions of MobileNet, with MobileNet V3 being the most recent.

In MobileNet V1[7], traditional convolution layers are replaced by depth-wise separable convolutions, which improves efficiency. Depth-wise separable convolutions factorize the standard convolution into a depthwise convolution that performs filtering, and a 1×1 convolution called a pointwise convolution that combines the output of the depthwise convolution. In a standard convolution layer, both filtering and combination are performed by using a single layer. Factoring them increases efficiency. In the second version of MobileNet, MobileNetV2, to increase the efficiency, a linear bottleneck and inverted residual structure

are introduced. The structure is defined by a 1×1 expansion convolution, followed by a depthwise convolution and a 1×1 projection layer.

MobileNetV3 [8] employs a platform-aware Neural Architecture Search (NAS) complemented by the NetAdapt algorithm. The NAS network search is used to find a target architecture from a seed architecture to optimize the efficiency of the network by reducing the latency and number of operations in each iteration. While NAS searches for a global architecture, the NetAdapt algorithm sequentially optimizes the layers of the architecture. The nonlinear Swish operation in the network is replaced by an efficient hard-Swish operation that utilizes the ReLU6 activation function. The squeeze and excite bottleneck layers are also included in the design. MobileNetV3 has two versions, MobileNetV3 large and MobileNetV3 small. Large and small values denote resource availability in the platform. Both MobileNetv3 demonstrated accuracy and latency improvements when used in image detection and segmentation tasks. The swish and h-swish (hard swish) functions are given by (1) and (2).

$$swishx = x\sigma(x) \tag{1}$$

$$h - swishx = x\frac{ReLU6(x+3)}{6} \tag{2}$$

Where, ReLU6 is a ReLU activation function with maximum output value 6 and σ is the sigmoid function.

2.3.2 Xception

Xception architecture [9] is used in computer vision. It is a deep convolution neural network architecture. It used depthwise separable convolutions. Depthwise separable convolutions are efficient alternatives of classical convolutions that reduce the time. The classical convolution is split into depthwise convolution and pointwise convolution, thereby reducing the number of operations, and increasing the efficiency. The architecture of the Xception contains 3 flows: entry flow, middle flow and, exit flow. The Xception architecture performs better than inception v3 [10, 11, 12] on Imagenet dataset by a small margin and outperforms inception v3 on large datasets by a large margin.

2.3.3 EfficientNet

Minqxing Tan et al. [13] demonstrated a method to scale width, depth and resolution of a CNN on a new baseline model architecture called EfficientNet. EfficientNet-B0 was

developed using the network architecture search that optimizes both the accuracy and the FLOPS of the model. Using the scaling methods proposed by Minqxing Tan et al. on the EfficientNet architecture, a family of EfficientNet models was obtained (EfficientNet B1-B7). EfficentNet-B7 demonstrated a top-1 accuracy of 84.3% on the ImageNet dataset and a top-1 accuracy of 91.7% on the CIFAR 100 dataset by using transfer learning. The EfficientNet models require high computational resources for training and has 43M parameters. The next version of EfficientNet addresses this problem. EfficientNet V2 optimizes training speed and parameter efficiency apart from accuracy and FLOPS of the model. The EfficientNet v2 model demonstrated a top-1 accuracy of 83.9% on the Imagenet dataset, while the number of parameters was 23M [14]. The EfficientNet V2 model that was obtained from network architecture search is also scaled to obtain EfficientNetV2-M/L models.

2.3.4 Comparing Pre-Trained Models

ImageNet dataset contains 1000 classes and is used to pre-train the models for image classification. Various models pretrained on this dataset are available in platforms such as TensorFlow hub. Therefore, using ImageNet dataset pre-trained models help to compare different models and choose an appropriate model for the application.

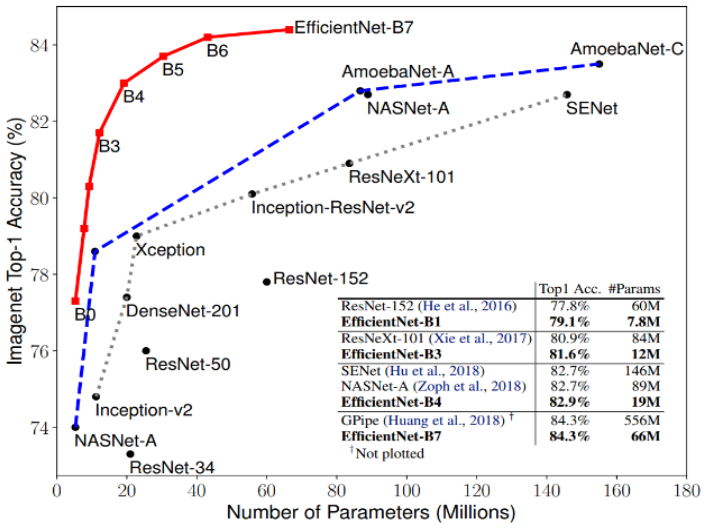

Figure 1. Graph showing accuracy of various models and number of parameters for each model (Tan et al. [13])

In Figure 1, the EfficientNet B7 model shows the highest accuracy of 84.3%. EfficientNet B7 has 66 million parameters, which makes it more difficult to train. The EfficientNet V2 model has 24 million parameters and an accuracy of 83.9%. The EfficientNet V2 model has fewer parameters and better accuracy than the EfficentNet B7

model. The MobileNet model has 4.2 million parameters and an accuracy of 70.6%. The EfficientNet V2 and MobileNet models are better when considering the accuracy and training efficiency.

2.3.5 Performance Metrics for the Models

Measures can be used to analyse the performance of the classification models. The measures that are chosen for the analysis task are accuracy, precision and recall [15]. These measures can be calculated as,

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{3}$$

$$Precision = \frac{TP}{TP+FP} \tag{4}$$

$$Recall = \frac{TP}{TP+FN} \tag{5}$$

Where,

TP = True positive count

FP = False positive count

TN = True negative count

FN = False negative count

3. Proposed Work

The proposed tool trains multiple pre-trained models on the target dataset and compares the models based on prediction accuracy to determine the best model for the target dataset. In addition, the proposed tool preprocesses the dataset. Figure 2 shows the architecture of the proposed tool.

The overall flow of the proposed tool is as follows: The target dataset is balanced (data balancing). The images in the balanced dataset are normalized and resized (pre-processing images) according to the requirements of each model. The models are trained on the preprocessed images using transfer learning. The accuracy of the model is determined by using a test dataset. The model with the highest accuracy is used for further predictions. Further subtopics in this section discuss the implementation of various modules of the proposed tool. These modules include data balancing, image pre-processing, model training, and model selection.

3.1 Data Balancing

Data balancing is applied to the input dataset to impose a balance on the number of images of each class. This step removes bias from any specific class of image. For data balancing, oversampling is used with augmentation methods, such as horizontal flipping, light shifting, and multi-cropping. In horizontal flipping, the image is flipped horizontally, producing a mirror image. Light shifting changes the RGB values of images, thereby changing the contrast by a certain value. Multi-cropping produces five crops in the original image. The five crops are taken from the top-left, top-right, bottom-left, bottom-right, and center of the image. The crop size will be 64 pixels less than the original size; thus, the cropped images will have a local spatial translation of the objects without missing important details. The use of multi-cropping on very small images may cause a loss of features. Therefore, flipping and light shifting are more important than multi-cropping.

In the augmentation method, an image is chosen randomly for a class, and any augmentation method is randomly chosen and applied to the image. The resulting augmented images are then added to the class. Augmentation is applied until a tolerating difference between the number of images in the current class and the highest number of images in all classes are achieved. It is given by (6).

$$\frac{n_C - n_H}{n_H} \leq \delta_D \tag{6}$$

Where,

n_C is the number of images in the augmenting class.

δ_D is the tolerating threshold.

n_H is the highest number of images in all the classes.

3.2 Image Pre-Processing

Image pre-processing steps are performed on the balanced dataset. The pre-processing steps are resizing, normalization and image centering. The images are resized according to the model input size, as different models require different input sizes. An image is a three-dimensional vector, containing three values corresponding to Red, Green, and Blue (RGB) values for each pixel with range [0,255]. These RGB values are normalized to the range [0,1]. Normalized inputs reduce the converging time of the deep learning models. Image centering is the technique in which the mean of the values is set to 0. This is achieved by subtracting each pixel value by the mean of all pixel values.

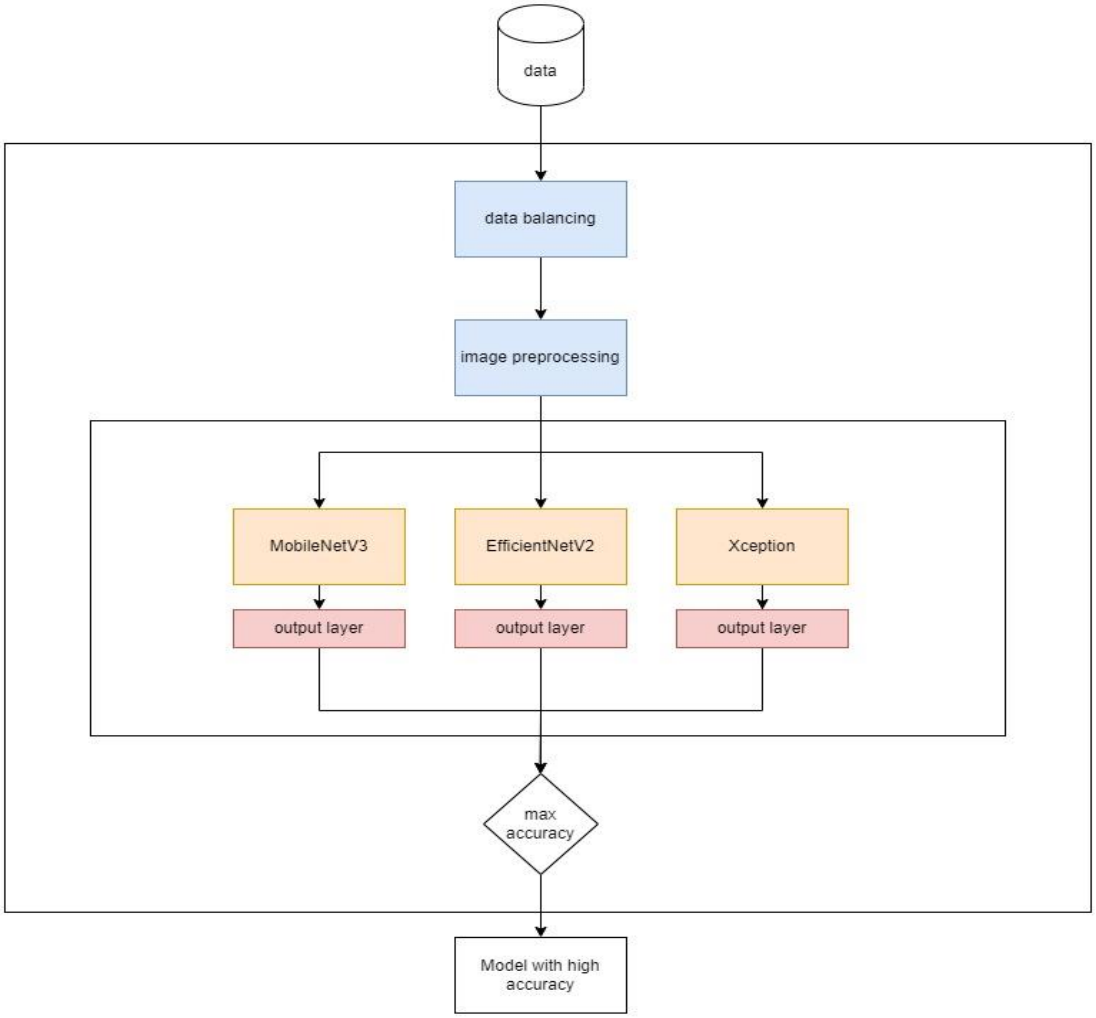

Figure 2. Architecture of the Tool Representing the Flow of the System

In fig.2, the data provided by the users first go through the automated data balancing step followed by the necessary pre-processing and finally used to train three different models which are pre-trained on ImageNet dataset. After training the model, the one that gives the best accuracy is selected and provided to the user.

3.3 Model training and selection

In model training, Xception, EfficientNet V2 B3 and MobileNet V3 models are trained on the same training dataset. Transfer learning is used to train the models. To implement transfer learning, an output layer with the number of nodes equal to the number of classes is added to the pre-trained models. During training, the output layer parameters are learned. The parameters of the pre-trained model remain unchanged. Table 1 lists the models used in training along with its input size. The model that produces the highest accuracy is selected for prediction in the target dataset.

Table 1. Models used along with its input size and pre-training dataset

Model	Image size	Pre-trained on
EfficientNet v2 imagenet21k ft1k b3	300 x 300	Imagenet 21k
MobileNet v3 large 100 224	224 x 224	Imagenet
Xception	299 x 299	Imagenet

3.4 Dataset description

3.4.1 Construction dataset

The dataset used for testing the accuracy of this solution contains images of objects related to construction. The dataset contains nine classes and 4624 images. The dataset is acquired by web scraping, followed by removing unsupported image formats using the 'pillow' library in python. Manual checking is performed on the dataset to ensure the consistency of all the images. As the dataset is web scraped, it is imbalanced and each image varies in size, enabling testing of the solution in various settings. Table 2 contains the nine classes included in the dataset along with their image instance counts. The uneven image distribution in each class is presented in Table 2.

Table 2. Number of images in each class of the unbalanced construction dataset

Class	Number of Instances
Bricks	531
Wood	506
Backhoe loader	642
Road roller	589
Cement	189
PVC pipes	454
Reinforced steel	483
Construction worker	568

Tower crane	662

This dataset was split in a ratio of 80:20 for training and testing. Eighty percent of the images are used to train the models, and each model is trained on the same images. Twenty percent of the images are used to analyze the accuracy of the trained models. The balanced construction dataset has 5958 images (obtained using the proposed tool), with 662 images for each class.

3.4.2 Minet 5640

The Minet dataset is used to create a deep learning model for identifying and classifying minerals. The dataset contains seven classes and 5640 images. From Table 3, it is evident that the dataset is unbalanced. The balanced Minet dataset (obtained using the proposed tool) contains 8259 images. Each class contains 1185 images in the balanced Minet dataset.

Table 3. Number of images in each class of the unbalanced Minet dataset

Class	Number of instances
Biotite	1070
Bornite	419
Chrysocolla	540
Malachite	998
Muscovite	344
Pyrite	1086
Quartz	1185

4. Results and Discussion

A machine with Intel Xenon Silver 8-core processor and Nvidia Tesla V100 16GB GPU is used to run the proposed tool. The proposed tool is tested on balanced and unbalanced construction dataset. Initially, the dataset is split at a ratio of 80:20. Eighty

percent of the dataset was used for training, with batch sizes of 512 and 10 epochs. The remaining twenty percent of the dataset is used to test the models. The model accuracies after training are presented in Table 4. As shown in Table 4, the MobileNet model has the highest accuracy of 81.80%, and EfficientNet V2 B3 has the lowest accuracy of 58.25%.

Table 4. Accuracy precision and recall of each model on the unbalanced construction dataset

Model	Accuracy (%)	Precision	Recall
EfficientNet v2 imagenet21k ft1k b3	58.25	0.41	0.58
MobileNet v3 large 100 224	81.8	0.78	0.80
Xception	67.7	0.62	0.71

The model accuracy for the balanced construction dataset is presented in Table 5. The training batch size is 512 and each batch was trained for 10 epochs. It is evident that the accuracy of the model improves when balancing the dataset. The accuracy of the MobileNet model improved by 14.03% for the balanced construction dataset. Whereas the EfficientNet V2 B3 model has the lowest accuracy of 76.04%.

Table 5. Accuracy precision and recall of each model on the balanced construction dataset

Model	Accuracy (%)	Precision	Recall
EfficientNet v2 imagenet21k ft1k b3	76.04	0.66	0.77
MobileNet v3 large 100 224	95.83	0.96	0.95
Xception	79.16	0.75	0.78

The proposed tool was tested on balanced and unbalanced Minet dataset. Initially, the dataset was split at a ratio of 80:20. Eighty percent of the dataset was used for training, with batch sizes of 512 and 10 epochs. The remaining twenty percent of the dataset was used to test the models. The same batch size is used for balanced and unbalanced Minet dataset. The model accuracies for the unbalanced Minet dataset and the balanced Minet dataset is presented in Table 6 and Table 7 respectively. For both the balanced and unbalanced Minet

dataset, the MobileNet model has the highest accuracy. The MobileNet model's accuracy increased on balancing the Minet dataset.

Table 6. Accuracy precision and recall of each model on the unbalanced Minet dataset

Model	Accuracy (%)	Precision	Recall
EfficientNet v2 imagenet21k ft1k b3	67.96	0.59	0.67
MobileNet v3 large 100 224	81.25	0.78	0.70
Xception	39.45	0.32	0.39

Table 7. Accuracy precision and recall of each model on the balanced Minet dataset

Model	Accuracy (%)	Precision	Recall
EfficientNet v2 imagenet21k ft1k b3	76.95	0.69	0.52
MobileNet v3 large 100 224	84.37	0.77	0.79
Xception	39.06	0.37	0.40

Table 8. GTM's accuracy, precision and recall for construction dataset and Minet dataset

Dataset	Accuracy (%)	Precision	Recall
Construction dataset	87.5	0.84	0.85
Minet dataset	70.3	0.74	0.70

Table 8 shows the performance of the model trained on GTM. The GTM model was trained for 10 epochs with batch size of 512 using the same training data used for the proposed tool. Comparing the model performance of the proposed tool and the GTM exported model, the proposed tool's model has better accuracy.

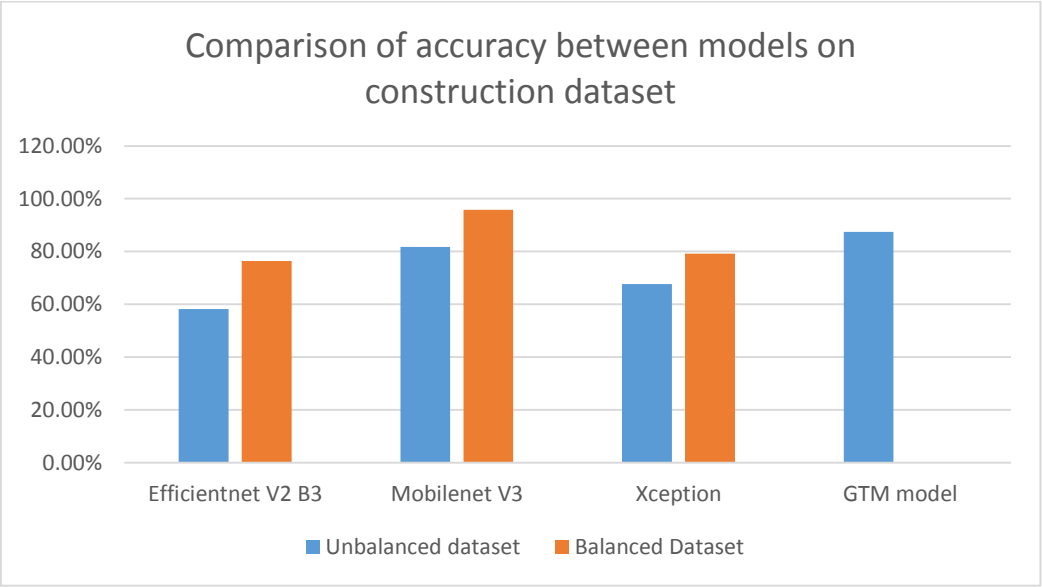

Figure 3. Graph depicting the accuracy of each model on both unbalanced and balanced construction dataset along with the accuracy of GTM model on construction dataset

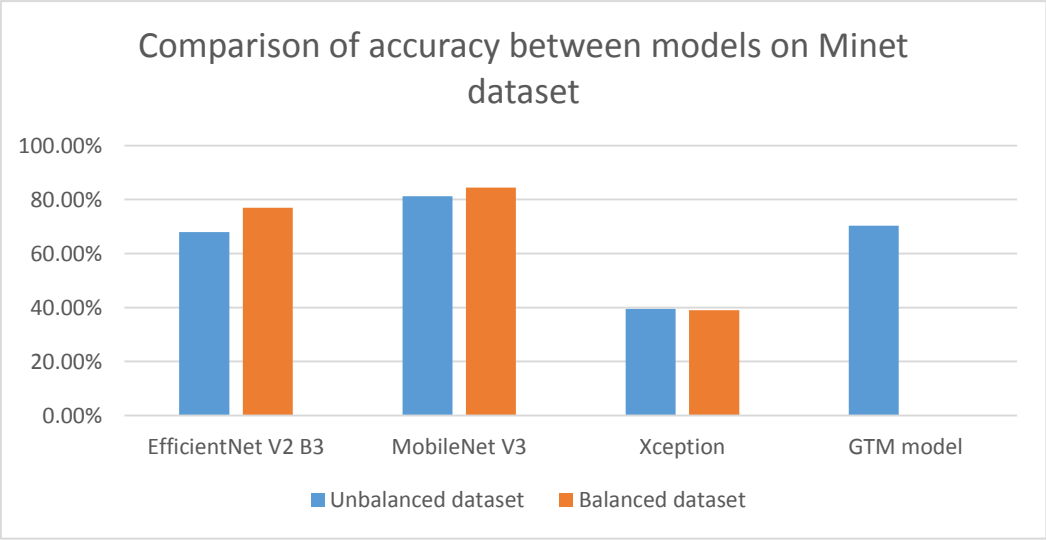

Figure 4. Graph depicting the accuracy of each model on both unbalanced and balanced Minet dataset along with the accuracy of GTM model on Minet dataset

5. Conclusion

In this paper, the implementation of a tool to automate various processes in the training of an image classification model is described. The different processes described are: 1) data balancing, where the distribution of number of images in each class is equalized to avoid bias to a particular class of images; 2) image pre-processing comprising the resizing, normalization, and image centering steps; and 3) model training, which trains EfficientNet V2-B3, Xception and MobileNet V3 pretrained models (pretrained on ImageNet or ImageNet 21k) and selects the model with the highest accuracy. Transfer learning is employed to train

the models on the target dataset. Testing the solution on an imbalanced construction dataset has showed the results of the implementation. It can be observed that the models produced better results after data balancing. Due to the automated pre-processing steps, the average increase in the accuracy is 14.55% in the construction dataset and 3.91% in the Minet dataset. Comparing to the GTM models, the tool produced model with 8.33% more accuracy on the construction dataset and model with 14.07% more accuracy on the Minet dataset. Thus, using the image classification model selector facilitates the creation of an effective image classification model for the target dataset. This reduces the workload of balancing the dataset and selecting a model from among multiple models used for image classification.

References

[1] P. Yogendra Prasad, Dr. Dumpa Prasad, Dr.D Naga Malleswari, Monali N. Shetty, Dr Neha Gupta. “Implementation of Machine Learning Based Google Teachable Machine in Early Childhood Education” (2022).

[2] Paula Branco, Lu´ıs Torgo, Rita P. Ribeiro. “A Survey of Predictive Modelling under Imbalanced Distributions” (2015).

[3] Tan, Mingxing, Quoc V. Le. "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks." (2019).

[4] Kotsiantis, Sotiris & Kanellopoulos, D. & Pintelas, P. “Handling imbalanced datasets: A review”. GESTS International Transactions on Computer Science and Engineering (2005). 25-36

[5] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, W. Philip Kegelmeyer. “SMOTE: Synthetic Minority Over-sampling Technique” (2002).

[6] Karl Weiss, Taghi M. Khoshgoftaar, DingDing Wang. “A survey of transfer learning”. DOI 10.1186/s40537-016-0043-6 (2016).

[7] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, Hartwig Adam. “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications” (2017).

[8] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, Quoc V. Le, Hartwig Adam. “Searching for MobileNetV3” (2019).

[9] François Chollet. “Xception: Deep Learning with Depthwise Separable Convolutions” (2017).

[10] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich. “Going deeper with convolutions.” (2015).

[11] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna. “Rethinking the inception architecture for computer vision.” (2015)

[12] C. Szegedy, S. Ioffe, V. Vanhoucke. “Inception-v4, inception-resnet and the impact of residual connections on learning.” (2016)

[13] Tan, Mingxing, Quoc V. Le. "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks." (2019).

[14] Mingxing Tan Quoc V. Le. “EfficientNetV2: Smaller Models and Faster Training”.

[15] Hossin, Mohammad & M.N, Sulaiman. “A Review on Evaluation Metrics for Data Classification Evaluations”. International Journal of Data Mining & Knowledge Management Process. 5. 01-11. 10.5121/ijdkp.2015.5201 (2015).

Author's Biography

N. Arulanand is a professor who has a diverse background of industry experience, teaching experience, and research publications. He has experience in Embedded Systems, IoT, Machine Learning and Image Processing. He has a strong track record of publishing research papers in these fields, which demonstrates his expertise and knowledge in the area. Dr. Arulanand N has spent a significant amount of time researching and developing new technologies and techniques related to embedded systems, IoT, machine learning, and image processing, which can be applied to various industries. With his industry experience, he brings a unique perspective to teaching, with a focus on practical applications of these technologies.

D. Kamalraj is an undergraduate student from PSG College of Technology, Coimbatore, Tamilnadu, India. He has experience in machine learning, deep learning, image processing, android development, and web development. He has also worked on full stack development projects.

B. Krishna Teja is an undergraduate student from PSG College of Technology. He has worked on projects related to image classification using deep learning. He has a keen interest in learning new technologies and developing applications.