Evaluation of Deep Learning Models using Benchmark Image Dataset

Manoj Gupta[1*], Anand Kumar Jain[2] and Dr. Amit Doegar[3]

[1] M.E.(CSE), Dept. of CSE, NITTTR, Chandigarh, India
[2] M.E.(CSE), Dept. of CSE, NITTTR, Chandigarh, India
[3] Associate Prof., Dept. of CSE, NITTTR, Chandigarh, India
manoj.cse20@nitttrchd.ac.in, manojmann88@yahoo.co.in

Abstract. Deep learning is a branch of artificial intelligence that is based on the idea that machines can learn from data, spot patterns, and make decisions with little help from humans. Deep learning models use decision trees, support vector machines (SVMs), k-Nearest neighbors, logistic regression, convolutional neural networks (CNNs), recurrent neural networks (RNNs), autoencoders, and transformers. Comparison of deep learning models YOLO, SSD, Faster RCNN, EfficientNet, ResNet50, and MobileNetV2 to determine which is best for a given situation. The fast RCNN method was used to make a classification framework that combined classification and feature extraction. CNNs are used to classify pictures of flowers, and a small SSD can be used for embedded detections. Compound scaling was used to make ResNet50, MobileNet, and Xception by scaling the baseline network to the right size. Xception uses depthwise separable convolutions to get state-of-the-art results on image classification tasks while using less memory and processing power. The EfficentNet model outperforms the ResNet50 and MobiileNetV2 models when the batch size changes from 16 to 64.

Keywords: ImageNet, EfficientNet, MobileNetV2, ResNet50, Custom dataset.

1 Introduction

Recent applications of computer vision to the industrial revolution have emerged. Deep learning is extensively used in the automation, robotics, healthcare, and monitoring sectors [1]. As a result of this accomplishment, which is most evident in applications such as object recognition, language processing, and image classification, deep learning has received the most attention. The market forecast anticipates exceptionally substantial development in the coming years. It is believed that the availability of powerful graphics processing units (GPUs) and a large number of datasets is one of the primary driving forces behind this [1]. Both necessities are currently easily accessible [4].

Machine learning is a technique of analysis of data that enables the creation of analytical models. It is a subfield of artificial intelligence based on the premise that machines can learn from data, identify patterns, and make decisions with minimal human input.

There are several types of machine learning:

Supervised learning: The training data are labeled in this form of machine learning system. The objective of the algorithm, given input data and the desired output, is to train a function that converts the input to the desired output.

Unsupervised learning: Due to the absence of labeled training data, unsupervised learning is a form of machine learning in which the algorithm must uncover data structure on its own.

Semi-supervised learning: For training purposes, this form of machine learning algorithm employs both labeled and unlabeled data.

Reinforcement learning: This type of machine learning algorithm acquires new skills by interacting with its environment and observing the positive or negative consequences of specific behaviours.

Deep learning is a subfield of machine learning that employs enormous datasets to train artificial neural networks to learn. It is based on the structure and function of the brain, specifically the connections between neurons.Deep learning algorithms learn and make decisions using multilayered artificial neural networks. Each layer processes the input data before passing it on to the following layer, which generates the output. Concealed layers are the layers that exist between the input and output layers.

Deep learning algorithms can be taught to recognise images and speech, comprehend natural language, and even perform games, among other tasks. They are exceptional pattern recognizers.Deep learning models are not the only machine learning models applicable to a wide range of applications. Several examples include:

Decision Trees: These models have a tree-like structure, with each node representing a feature, each branch representing a decision, and each leaf representing the outcome. Classification and regression problems can be solved using decision trees.

Random Forest: This assortment of decision trees is known as a "random forest." Combining the predictions of multiple decision trees, Random Forest models are created following training. Frequently more accurate than a single decision tree, they are frequently used for classification and regression, among other tasks.

Support Vector Machines (SVMs): These classification models are founded on the concept of locating a hyperplane that most effectively separates the various classifications.

k-NN is a straightforward, nonparametric classification and regression technique. In order to generate predictions for new data, the model is trained on a dataset and then searches for the K-training examples that are most similar to a new data point.

Linear Regression: A scalar response and one or more explanatory variables are modelled using a linear approach. It is employed to forecast continuous values using one or more predictor variables (dependent variables).

Logistic Regression: A logistic model is fitted using the statistical technique of logistic regression. A set of independent variables is employed to predict a binary outcome (1/0, Yes/No, True/False).

There are numerous categories of deep learning models, including:

CNNs are employed for image and video recognition tasks. They are designed to process data with a topology resembling a grid, such as an image.

Recurrent neural networks (RNNs) are utilised to predict time series and process natural language using sequential data.

Autoencoders models can learn to represent data in a lower-dimensional space via unsupervised learning, which is typically employed for dimensionality reduction or learning features.

Generative Adversarial Networks (GANs) are necessary to train two neural networks, a discriminator and a generator, for these unsupervised learning models to function. The generator creates false data, while the discriminator attempts to distinguish between bogus and actual data.

Transformer is novel neural network architecture is effective in a variety of natural language processing applications. It excels at tasks that require sequential information and is based on techniques for self-attention control.

The deep learning models YOLO, SSD, Faster RCNN, EfficientNet, ResNet50, and MobileNetV2 are compared in this review article [2]. SSD is the first technique compared in the present study. SSD enhances the end network with multiple layers of features and simplifies detection [3]. Accelerated RCNN is a unified, quicker, and more accurate approach to object recognition based on convolutional neural networks. Joseph Redmon conceived of the end-to-end YOLO network [3]. According to the findings of the researchers, the compound scaling procedure can enhance the EfficientNet model's precision and efficacy. Resnet can incorporate a network variant that has been pre-trained using more than one million images from the ImageNet database. A trained network that can accept images as large as 224 by 224 pixels can classify a mouse, a keyboard, a stylus, and several other animals into one of 1000 distinct object categories. MobileNet's depth-wise separable convolution reduces the size and intricacy of network models, making it suitable for mobile devices and other low-powered computing systems.

The Microsoft COCO and ImageNet datasets were used as a common factor of analysis [16] to compare the relative efficacy of the aforementioned methods, which all use distinct architectures. By comparing how well various algorithms perform on the same dataset, it is possible to determine which method for identifying objects is optimal in a given circumstance. By comparing the results, you can also determine how each algorithm is distinct from the others.

2 Background

The rapid RCNN method was utilised to create a framework for classification that incorporated classification and feature extraction. Object detection has garnered substantial research interest in recent years. The regression method is utilised to resolve the object detection issue. Ross Girshick developed the Fast RCNN model, which has become a popular object detection method [3]. This study covered RCNN, Rapid RCNN, and Accelerated RCNN fundamentals. The CNN technique is utilised for target detection. Rapid RCNN requires training time nine times less than RCNN. The accuracy of rapid RCNN and accelerated RCNN is identical. YOLO is an additional detection network [8]. The process integrates object recognition with deep learning and operates at 5-7 frames per second (fps), according to the research [5]. MicroSD was developed to facilitate the identification of real-time embedded objects. This work develops a system for detecting and identifying moving objects [7] using CCTV (Closed Circuit Television) cameras by integrating CNN with background reduction. The region proposal network, a network template utilised by the Faster RCNN method, is created by combining the proposal isolation region and a small component of the Fast RCNN network (RPN). The researchers [11] propose Micro SSD, a deep convolutional neural network for single-shot identification. Tiny SSD is 2300 bytes smaller than Tiny YOLO, which is its greatest strength.

The feature extraction and bounding box prediction of the YOLO architecture [6] utilised in this investigation were based on the method described here. In our research [10], the work presented here helped us determine how to analyse and train the SSD model. According to the findings of this investigation [9], a small SSD can be used for embedded detections.

It can be challenging to locate blossoms using deep learning due to the variety of flower shapes, sizes, and colours. Deep learning models, on the other hand, have been demonstrated to be exceptionally excellent at classifying images, so they can be used to locate and classify images of flowers.Using a convolutional neural network (CNN) to classify images of flowers is one method to discover flowers using deep learning. CNNs are a form of deep learning model whose ability to recognise image patterns makes them ideally suited for image classification tasks.To train a CNN for floral detection, flower images are necessary. Each image should be labelled with the flower species it depicts. Once the information has been produced, CNN can be trained on it. The trained model can then be utilised to classify images of new flowers.

Utilizing trained models such as ResNet, InceptionV3, etc., are alternative strategies. By training the final layer of the pre-trained model with our own dataset, we can customise these models for a specific purpose, such as flower detection. The effectiveness of the model will depend on the size and variety of the dataset as well as its usability. With a large and diverse dataset and a well-designed CNN, it is possible to accurately locate flowers. In the case of rare and peculiar plant species, however, the accuracy may be low. If you wish to continue with this task, I recommend researching popular datasets for flower detection, such as the Oxford Flowers dataset, as well as libraries such as Tensorflow and Keras, which provide pre-trained models and tools for training and evaluating models.

Fang et al. [35] proposed an enhanced, faster R-CNN that employs the HOG and sliding window techniques to recognise real-time objects such as labourers and heavy machinery. Their detection of excavators is 95% accurate, while their employee identification is 91% accurate. The R-CNN and ZF Network employed by Cheng and Wang's [36] automated procedure for locating a defect in a wastewater conduit are accelerated. Three thousand images extracted from CCTV surveillance recordings were used to train the model. The model's mAP was 83%. Vesal et al. [37] proposed the Faster R-CNN model with ResNet50 as the backbone network and U-Net architecture for lesion localization. Both the PH2 datasets and the ISBI 2017 challenge were incorporated during the model's training and evaluation. The average DC and JI scores during the investigation were 93.4% and 88%, respectively. Al-Azzoa et al. [38] developed models for human health-related detection utilising SSD MobileNet and Accelerated R-CNN ResNet. The models were evaluated using the publicly available 3D action recognition dataset. The SSD model had a mAP of 95.8%, while the R-CNN model had a mAP of 93.8%. Ren et al. [39] designed a modified Faster R-CNN with ResNet50 for detecting minute objects using optical remote sensing and images. During accelerated R-CNN training, random rotation augmentation was used. Detailed feature maps were generated using RPN anchors with modifications. In addition, the context information associated with an object proposal during training enhanced the performance of small object detection.

3 Methodology

A retrained image classifier is a machine learning model that is trained on a new set of data using a previously trained model as a starting point. This procedure, known as fine-tuning, enables the model to learn new features and adapt to new data while utilising the knowledge acquired through training. When there is a limited quantity of labelled data available, this method can be advantageous because it enables the model to learn from past experience and improve its performance on the new task.

Image classification models incorporate millions of parameters. To train them from scratch, a substantial quantity of labelled training data and processing power are required. Transfer learning is a technique that speeds up this process by transferring into a new model a portion of an existing model that has been trained on a similar task.

This technique demonstrates how to create a Keras model for classifying five distinct types of flowers using a pre-trained TF2 saved model from TensorFlow Hub for visual feature extraction, trained on the substantially larger and more versatile ImageNet dataset. In addition to the newly added classifier, it is possible to train (or "tune") the feature extractor.

In the most recent phase of the search for the optimal algorithm and data set, competitors have utilised the most renowned and highly regarded deep learning architectures and data sets. Microsoft COCO and ImageNet are the two most frequently used datasets for this purpose that achieve the highest levels of precision and accuracy.

The Custom Dataset contains tens of thousands of images of flowers. This dataset comprises a total of 3,670 images. Where relative images of flowers can be found in each directory. These five subdirectories, one for each class, make up the flower's dataset. This bespoke dataset included daisies, sunflowers, roses, tulips, and dandelion flowers.

Custom Dataset Retraining Methodology includes following steps:

3.1 Input Custom Flower Dataset (TF).

In this step we give input of custom flower dataset. It is URL as given. https://storage.googleapis.com/download.tensorflow.org/example_images/flower_photos.tgz

3.2 Take Image Features specified vectors (ImageNet) Models.

In this step first we import tensorflow setup and select the TF2 Saved Model module to use.

SSD

Object identification models, such as Accelerated RCNN or YOLO, have a substantially superior object-detection technique than SSD due to their significantly delayed processing speed. Before the SSD was created, a number of attempts were made to speed up the detector by altering each stage of its operation. Instead of merely modifying an existing model for object detection, researchers determined that an entirely new model was required. This resulted in the development of the SSD model [8] However, if these adjustments have a significant impact on performance, they will reduce the detection's precision.

SSD is just as accurate as models that use bounding box hypotheses, but pixels and features are not resample. By consolidating processing into a single network, the resampling portions of generating pixels and proposals are eliminated. This greatly simplifies its use compared to methodologies that require object proposals. SSD is simple to train and incorporate because one of the system's responsibilities is to locate objects. [8]

Inception V2

Inception is yet another basic architecture that can serve as a detection model's foundation. The initial paper by Szegedy, Inception V1, aimed to enhance CNN performance without increasing network size or processing costs [29]. Their decision to employ filters of different sizes at the same network level allows them to derive acceptable features regardless of the object's size. Additional 1x1 convolutions were implemented in order to reduce the number of input channels and the associated computing needs. With the release of Inception V2 [28], the issue of "representational congestion" was addressed, which occurs when convolutions modify the input dimensions disproportionately, resulting in a substantial loss of information. In lieu of delving deeper into a network, which would lead to a reduction in dimensions, their strategy consisted of expanding the filters [30].

Faster RCNN

The abbreviation RCNN stands for region-based convolutional neural networks. [20] This method combines region suggestions with high-capacity CNNs for segmenting and identifying objects. Following is the algorithm for the conventional R-CNN technique: [] The provided image is utilised with a selective search technique to generate multiple potential region recommendations. This procedure generates a significant number of prospective areas during the initial subsegmentation. Then, contiguous zones are connected using a greedy strategy to construct larger regions. These localities are included in the remaining regions' recommendations. The CNN component transforms the suggestions into a vector with discrete properties [15]. A SVM receives the collected features to identify the intriguing objects in the proposal (Support Vector Machine).

Yolo v3

You Only Observe Once (YOLO) is currently one of the most successful and accurate object detection systems [9] available. It was constructed using the newly customised and modified Darknet architecture [25]. Google Net influenced the initial iteration by sampling the image and predicting it using tensor technology as precisely as possible. The Region of Interest (ROI) of the Faster R-CNN network serves as the premise for constructing the tensor. ROI is aggregated and merged to facilitate the analysis and reduce the number of distinct computations. The architecture of the second generation consisted of only 30 convolutional layers, 19 of which were derived from the DarkNet-19 dataset, and 11 layers for detecting actual items or objects in natural environments using the COCO dataset and metrics [21]. Despite issues with photos containing tiny objects and pixels, the detection was still quite rapid and more accurate. Nonetheless, the third version of YOLO has demonstrated to be the most effective and accurate [12] due to its high degree of accuracy. The multilayered design has also improved the detection's accuracy.

EfficientNet

Mingxing Tan and Quoc V. Le of Google Research and Brain discussed the EfficientNet model in their paper titled "Rethinking Model Scaling for Convolutional Neural Networks." This article was presented during the 2019 International Conference on Machine Learning. These researchers scaled the model and figured out how to improve performance by harmonising the network's resolution, depth, and scope. Using this observation as a jumping off point, they proposed a novel scaling strategy [34] that increases the network's depth, scope, and resolution appropriately.

Scaling

To scale the dimensions of the network, the researchers employed a technique known as compound scaling. Using a specified resource constraint, the grid search method was used to determine the relationship between the various scaling dimensions of the baseline network. Scientists determined the appropriate scaling factors for each dimension requiring expansion using this method. Utilizing these coefficients, the baseline network was scaled to the required dimensions. Initially, the researchers automated the construction of neural networks using a neural architecture search to create a baseline network.

ResNet50

ResNet-50 employs a neural network with 50 convolutional layers (48 layers of convolution, one layer of MaxPool, and one layer of average pool). The residual blocks from artificial neural networks (ANNs) are layered to form a residual neural network (ResNet).

The design of ResNet50 is divided into four phases. The network is able to receive images with heights, widths, and channel widths that are multiples of 32 in both dimensions [22]. The input quantity is multiplied by three for the sake of clarity. [17] Each architecture of ResNet employs 33 kernel sizes for maximal pooling and 77 kernel sizes for initial convolution. Stage 1 of the network comprises of three sectors with three layers each. The stage 1 block kernels are 64, 64, or 128 kilobytes in size during each of the three phases of convolution.

MobileNet

MobileNet is a CNN architectural paradigm [23] for mobile vision and image classification. MobileNet is distinct from other models in that its operation and application of transfer learning require comparatively few computational resources. It is well-suited for mobile devices, embedded systems, and personal computers because it does not require a graphics processing unit and has a low computing efficiency. In addition, because web browsers have computational, visual processing, and storage limitations, they are the most appropriate platforms for it.

MobileNets, which use depth-wise separable convolutions to construct lightweight deep neural networks, are recommended for mobile and embedded vision applications.

Xception

Francois Chollet created Xception in 2017 as a convolutional neural network architecture for image classification tasks. It is an expansion of Google's Inception architecture, which was introduced in 2014. The primary difference between these two architectures is that Xception employs depthwise separable convolutions as opposed to conventional convolutions.

Each element of the output tensor is calculated as the dot product of a filter and the input window in a typical convolution. The filters are applied in sliding window fashion to the input tensor. However, depthwise separable convolutions employ both pointwise and depthwise filters. Pointwise filters are employed to integrate the output of depthwise filters across channels, while depthwise filters are applied independently to each channel of the input tensor.

Using depthwise separable convolutions, the Xception model achieves state-of-the-art performance in image classification tasks while consuming significantly less memory and processing capacity. The pre-training of Xception began with a massive collection of images, which were then supplemented with images that were better adapted to the task. This process of fine-tuning allows the model to account for the unique characteristics of the new dataset. This increases the model's precision.

This experiment used six models of the feature vector URL. Which is EfficientNet_b7, Res-Net50v2, InceptionV2, InceptionV3, MobileNetV1, and MobileNetV2.

3.3 Choose Different Batch Size.

In this step we choose four batch size. Which is 16, 32, 48 and 64 batch size.

3.4 Training the Models.

In this step we split dataset in training and testing dataset and trained on 25 epochs.

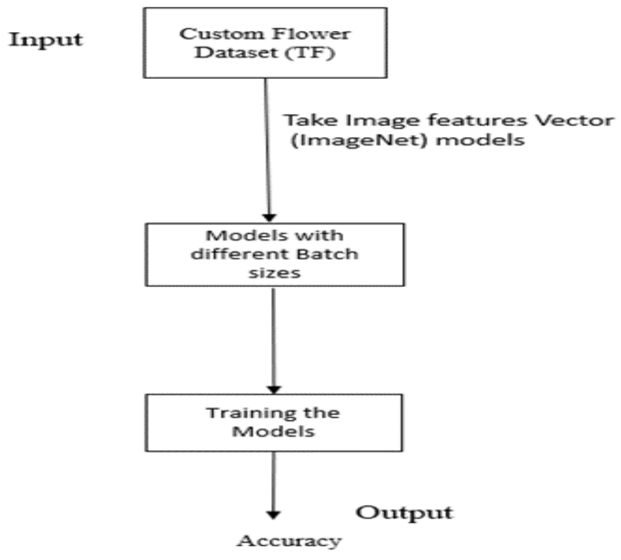
Input
Custom Flower Dataset (TF)
Take Image features Vector (ImageNet) models
Models with different Batch sizes
Training the Models
Output
Accuracy

Fig. 1. Flow chart of proposed methodology

3.5 Accuracy output.

In this step we take reading of accuracy as output for further analysis.
Accuracy= (TP+TN) / (TP+FN+FP+TN)
Recall (Sensitivity) = TP / (FN+TP)
Precision = TP / (FP+TP)
Specificity = TN / (TN+FP)
- where TP- True Positive, FP- False Positive, FN- False Negative and FP- False Positive values.

4 Results and Discussion

4.1 Experimental setup

Hardware

The computer includes a 1.8GHz Intel Core i5 and 8th Generation processor, a 1 TB hard drive, a 256 GB SSD (solid-state drive), and 8 GB DDR5 RAM (random access memory).

Software

Utilizes the Google Colab configuration software. It has 12.7 Gigabytes of RAM, but only 3.54 GB are typically in use. Additionally, it offers 107.7 GB of storage space, of which 74.4 GB were utilised for training and validation datasets. Google Colab's synthetic GPU is the hardware accelerator.

4.2 Experimental Results

In this experiment, there are various phases for flower image prediction like pre-processing, Region of Interest (ROI) creation, flower image detection, and flower image prediction, for verification we used the retraining image classifier as it gives better results. After that, we found that outcomes are improving: accuracy, precision, and recall. We use roses, sunflowers, tulips, dandelions, and daisy flower images as datasets in the implemented work. We used a ratio of 80:20 for training and testing. Images are in .jpg/.jpeg format which is extracted from different flowers using the bounding box. These images have a resolution of 320*232 pixels. The implemented technique is matched with the existing technology in terms of accuracy.

Table 1. Table captions should be placed above the tables.

Models	**Batch 16 (%)**	**Batch 32 (%)**	**Batch 48 (%)**	**Batch 64 (%)**
EfficientNet	94.58	94.74	94.58	94.03
ResNet50	86.53	89.20	89.31	90.62
InceptionV2	96.25	95.74	95.28	94.60
InceptionV3	95.97	95.88	95.69	94.60
MobileNetV1	89.72	91.76	91.53	92.05
MobileNetV2	92.64	93.04	93.06	92.90

EfficientNet

Accuracy in this model rises from 16_batch to 32_batch. This is a 0.16 percent increase in precision. The accuracy then drops from 32_batch to 64_batch. This is a 0.71% decline in precision. The EfficientNet model has an accuracy of approximately 94.74 percent.

ResNet50

Accuracy in this model increases from 16_batch to 64_batch. This is a 3.91 percent increase in precision. The ResNet50 model has an accuracy of up to 90.62 percent.

InceptionV2

The accuracy decreases from 16_batch to 64_batch in this model. This is a 1.65% decrease in precision. The inceptionV2 model has an accuracy of approximately 96.25 percent.

InceptionV3

The accuracy decreases from 16_batch to 64_batch in this model. This is a decrease in accuracy of 1.37 percent. The inceptionV3 model achieves an accuracy of approximately 95.97%.

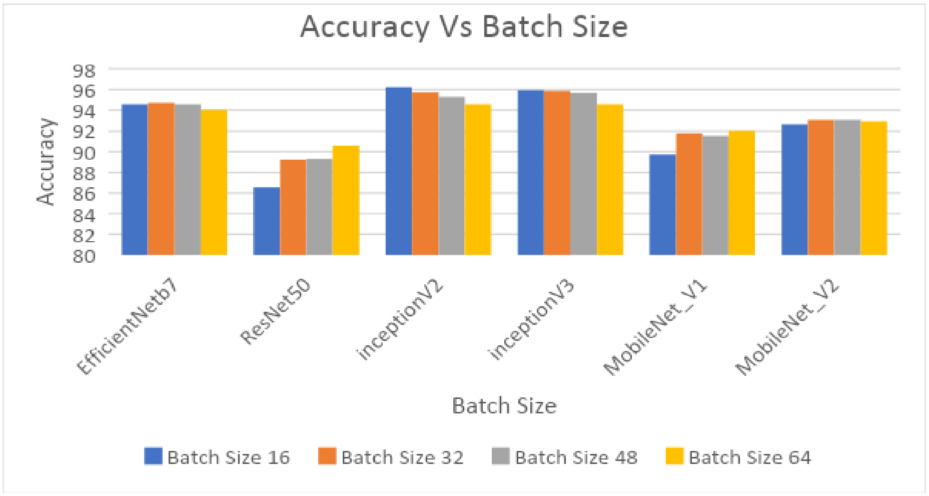

Fig. 2. Comparison of Accuracy with Batch Size.

MobileNetV1

The accuracy increases from 16_batch to 32_batch. This is a 2.04% increase in precision. After this, the accuracy degrades from 32_batch to 48_batch. This is a 0.23% decrease in precision. After this, however, the accuracy increases from 48_batch to 64_batch. This is a 0.52 percent increase in precision. The MobileNetV2 model has an accuracy of approximately 92.05 percent.

10
MobileNetV2

The accuracy of this model increases from 16_batch to 48_batch. This represents an increase in accuracy of 0.39%. The accuracy then decreases from 48_batch to 64_batch. This is a 0.16 percent reduction in precision. The MobileNetV2 model has an accuracy of approximately 93.06%.

Table 2. Comparison of our Proposed Models to Previous Models.

Model	**Dataset**	**Accuracy (%)**
Previous Model (VGG 16 transfer (Yong Wu et al. [77]))	Flower Dataset (TF)	83.53
Previous Model VGG 19 transfer (Yong Wu et al. [77])	Flower Dataset (TF)	84.71
Previous Model VGG 16 (Gadkari et al. [78])	Flower Dataset (TF)	87.95
Previous Model VGG 19 (Gadkari et al. [78])	Flower Dataset (TF)	88.57
Previous Model MobileNet (Wang et al. [79])	Flower Dataset (TF)	82.87
Previous Model Ensemble (Wang et al. [79])	Flower Dataset (TF)	91.81
Previous Model LeNet (BR.Mete et al. [80]) (Same Flower)	Oxford 102	57.73
Previous Model InceptionResNetV2 (Bozkurt et al. [81])	Kaggle flower dataset (Same dataset)	92.25
Proposed Model (EfficientNet)	Flower Dataset (TF)	94.74
Proposed Model (ResNet50)	Flower Dataset (TF)	90.62
Proposed Model (InceptionV2)	Flower Dataset (TF)	96.25
Proposed Model (InceptionV3)	Flower Dataset (TF)	95.97
Proposed Model (MobileNetV1)	Flower Dataset (TF)	92.05
Proposed Model (MobileNetV2	Flower Dataset (TF)	93.06

Our Proposed models were compared with VGG-16 transfer with 83.53%, VGG-19 Transfer with 84.71% (Yong Wu et al. [77])), VGG-16 with 87.95%, VGG-19 with 88.57% (Gadkari et al. [78]), Ensemble model with 91.81%, MobileNet model with 82.87% (Wang et al. [79]), LeNet with 57.73% (BR.Mete et al. [80]), InceptionResNetV2 with 92.25% (Bozkurt et al. [81]) accuracies. In this comparison, our proposed models were good compared to other models, which were EfficientNet with 94.74%, ResNet50 with 90.62%, InceptionV2 with 96.25%, InceptionV3 with 95.97%, MobileNetV1 with 92.05%, MobileNetV2 with 93.06% accuracy. Our Proposed Model is more accurate in Rose flower detection compared to other models.

5 CONCLUSION

Object detection is essential for the advancement of real-time technologies like autonomous vehicles. In this assessment, the most advanced CNN-based object detection algorithms were compared. Daily online publication of hundreds of thousands of photographs necessitates object detection [42]; without it, examining them would be difficult. Compared to ResNet50 and MobileNet, the EfficientNet model is more accurate with custom flower datasets than ResNet50 and MobileNet. Different sample sizes yield varying degrees of precision across all models in this manner. These topics may be investigated in greater depth in future studies.

References

1. Wu, Xiongwei, Doyen Sahoo, and Steven CH Hoi. "Recent advances in deep learning for object detection." Neurocomputing 396 (2020): 39-64.
2. I. Goodfellow, Y. Bengio, and A. Courville. " Deep Learning." MIT Press, 2016. http://www.deeplearningbook.org.
3. Palop JJ, Mucke L, Roberson ED. "Quantifying biomarkers of cognitive dysfunction and neuronal network hyperexcitability in mouse models of Alzheimer’s disease: depletion of calcium-dependent proteins and inhibitory hippocampal remodeling." In: Alzheimer’s Disease and Frontotemporal Dementia. Humana Press, Totowa, NJ; 2010, p. 245–26
4. Tosun, Selman, and Enis Karaarslan. "Real-Time Object Detection Application for Visually Impaired People: Third Eye." 2018 International Conference on Artificial Intelligence andData Processing (IDAP). Ieee, 2018.
5. Ren S, He K, Girshick R, Sun J. "Faster r-cnn: Towards real-time object detection with region proposal networks. " IEEE Trans Pattern Anal Mach Intell. 2016;39(6):1137–49.
6. Ding S, Zhao K. "Research on daily object detection based on deep neural network." IOP Conf Ser Mater Sci Eng. 2018;322(6):062024.
7. Kim C, Lee J, Han T, Kim YM. "A hybrid framework combining background subtraction and deep neural networks for rapid person detection." J Big Data. 2018;5(1):22.
8. Redmon J, Divvala S, Girshick R, Farhadi A. You only look once: Unified, real-time object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2016, pp. 779–788
9. Ahmad T, Ma Y, Yahya M, Ahmad B, Nazir S. Object detection through modifed YOLO neural network. Scientifc Programming, 2020.
10. Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu CY, Berg AC. Ssd: single shot multibox detector. In: European conference on computer vision. Cham: Springer; 2016, p. 21–37.
11. Womg A, Shafee MJ, Li F, Chwyl B. Tiny SSD: a tiny singleshot detection deep convolutional neural network for real-time embedded object detection. In: 2018 15th conference on computer and robot vision (CRV). IEEE; 2018, p. 95101
12. Chen W, Huang H, Peng S, Zhou C, Zhang C. YOLO-face: a real-time face detector. The Visual Computer 2020:1–9 IEEE, 2018.
13. Mitchell T. Machine learning. New York: McGraw Hill; 1997.
14. Schulz H, Behnke S. Deep learning. KI-Künstliche Intelligenz. 2012;26(4):357–63.
15. Khan A, Sohail A, Zahoora U, Qureshi AS. A survey of the recent architectures of deep convolutional neural networks. Artif Intell Rev. 2020;53(8):5455–516.

16. Chen, Xinlei, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C. Lawrence Zitnick. "Microsoft coco captions: Data collection and evaluation server." arXiv preprint arXiv:1504.00325 (2015).
17. Ranzato MA, Huang FJ, Boureau YL, LeCun Y. Unsupervised learning of invariant feature hierarchies with applications to object recognition. In: 2007 IEEE conference on computer vision and pattern recognition. IEEE; 2007, p. 1–8.
18. Krizhevsky A, Sutskever I, Hinton GE. Imagenet classifcation with deep convolutional neural networks. Adv Neural Inf Process Syst. 2012;25:1097–105.
19. Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556; 2014.
20. Girshick R. Fast r-cnn. In: Proceedings of the IEEE international conference on computer vision; 2015, p. 1440–8.
21. Redmon J, Farhadi A. Yolov3: an incremental improvement. arXiv preprint arXiv:1804.02767; 2018
22. Zhao ZQ, Zheng P, Xu ST, Wu X. Object detection with deep learning: a review. IEEE Trans Neural Netw Learn Syst. 2019;30(11):32123232
23. Zeiler MD, Fergus R. Visualizing and understanding convolutional networks. In: European conference on computer vision. Cham: Springer, 2014, p. 818–33
24. Google Colaboratory. https://colab.research.google.com/ notebooks/intro.ipynb.
25. LabelImg. https://github.com/tzutalin/labelImg.
26. TensorFlow model. https://www.tensorflow.org
27. Android studio. https://developer.android.com/android-studio/download
28. Chen, Xinlei, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C. Lawrence Zitnick. "Microsoft coco captions: Data collection and evaluation server." arXiv preprint arXiv:1504.00325 (2015).
29. Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Rabinovich A. Going deeper with convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2015, p. 1–9.
30. Szegedy, Christian, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. "Rethinking the inception architecture for computer vision." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2818-2826. 2016.
31. COCO Detection Evaluation. http://cocodataset.org/ #detection-eval.
32. Everingham, Mark, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. "The pascal visual object classes (voc) challenge." International journal of computer vision 88, no. 2 (2010): 303-338.
33. Python. https://www.python.org/downloads/
34. Tan, Mingxing, and Quoc Le. "Efficientnet: Rethinking model scaling for convolutional neural networks." International conference on machine learning. PMLR, 2019.
35. Fang, Weili, et al. Automated detection of workers and heavy equipment on construction sites: A convolutional neural network approach. Advanced Engineering Informatics 37 (2018): 139-149.
36. Cheng, Jack CP, and Mingzhu Wang. "Automated detection of sewer pipe defects in closed-circuit television images using deep learning techniques." Automation in Construction 95 (2018): 155-171
37. Vesal, Sulaiman, et al. A multi-task framework for skin lesion detection and segmentation. OR 2.0 Context-Aware Operating Theaters, Computer Assisted Robotic Endoscopy, Clinical Image-Based Procedures, and Skin Image Analysis. Springer, Cham, 2018. 285-293.
38. Al-Azzoa, Fadwa, Arwa Mohammed Taqia, and Mariofanna Milanovab. Human related-health actions detection using Android Camera based on TensorFlow Object Detection API. International Journal of Advanced Computer Science and Applications 9.10 (2018).
39. Ren, Yun, Changren Zhu, and Shunping Xiao. Deformable faster r-cnn with aggregating multi-layer features for partially occluded object detection in optical remote sensing images. Remote Sensing 10.9 (2018): 1470.
40. Fu, Longsheng, et al. "Kiwifruit detection in field images using Faster R-CNN with ZFNet." IFAC-PapersOnLine 51.17 (2018): 45-50.
41. Tian, Zhi, et al. "Fcos: Fully convolutional one-stage object detection." Proceedings of the IEEE/CVF international conference on computer vision. 2019.

42. Zheng, Yang-Yang, et al. "CropDeep: the crop vision dataset for deep-learning-based classification and detection in precision agriculture." Sensors 19.5 (2019): 1058.
43. Yu, Yang, et al. "Fruit detection for strawberry harvesting robot in non-structural environment based on Mask-RCNN." Computers and Electronics in Agriculture 163 (2019): 104846.
44. Qin, Wei, et al. "Flower species recognition system combining object detection and attention mechanism." International Conference on Intelligent Computing. Springer, Cham, 2019.
45. Lei, Xusheng, and Zhehao Sui. "Intelligent fault detection of high voltage line based on the Faster R-CNN." Measurement 138 (2019): 379-385.
46. Shao, Faming, et al. "Improved faster R-CNN traffic sign detection based on a second region of interest and highly possible regions proposal network." Sensors 19.10 (2019): 2288.
47. Zhang, Yiqi, et al. "The significance of incorporating unidentified vessels into AIS-based ship emission inventory." Atmospheric environment 203 (2019): 102-113.
48. Y. Liu, F. Tang, D. Zhou, Y. Meng and W. Dong, "Flower classification via convolutional neural network," 2016 IEEE International Conference on Functional-Structural Plant Growth Modeling, Simulation, Visualization and Applications (FSPMA), 2016, pp. 110-116, doi: 10.1109/FSPMA.2016.7818296.
49. Abu, Mohd Azlan, et al. "A study on Image Classification based on Deep Learning and Tensorflow." International Journal of Engineering Research and Technology 12.4 (2019): 563-569.
50. Puranik, P., et al. "Strawberry flower and fruit detection using deep learning for developing yield prediction models." Precision agriculture'21. Wageningen Academic Publishers, 2021. 1137-1149.
51. Solanki, Arun, and Tarana Singh. "Flower Species Detection System Using Deep Convolutional Neural Networks." Emerging Technologies for Computing, Communication and Smart Cities. Springer, Singapore, 2022. 217-231.
52. Sun, Kaiqiong, et al. "Apple, peach, and pear flower detection using semantic segmentation network and shape constraint level set." Computers and Electronics in Agriculture 185 (2021): 106150.