JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

FRUIT QUALITY CLASSIFICATION APPLICATION USING AN ARTIFICIAL INTELLIGENCE ALGORITHM

Evaluating and categorizing fruits based on their quality like size,color,ripeness,freshness or presence of defects.

[1]Sharada HP, [2]Dr. Sandeep Bhat

[1]Mtech Student, [2]Associate Professor, Computer Science and Engineering
[1]Computer Science and Engineering
[1]Srinivas Institute of Technology, Valachil, Mangaluru -575001 Karnataka, India

Abstract: The assessment of fruit quality in markets is becoming more difficult as mainly due to the demanding physical labor it requires. "This work introduces a low-cost method for assessing fruit quality by combining camera-based image capture with AI-driven classification." The system is designed to operate effectively in real-world environments. Fruit detection is primarily carried out using the "You Only Look Once" (YOLO) V3 algorithm. Designated fruits are continuously tracked based on image characteristics such as size, height, fressness and width, with quality assessment occurring throughout the process. A switching gap mechanism is employed to distinguish between different quality grades of fruit. This is optimized for round fruits, including apples, bananas, cucumbers, oranges, and lemons, and incorporates a newly developed image processing approach. Additionally, a graphical user interface (GUI) is provided to facilitate control, data collection, model evaluation, and system monitoring, thereby enhancing the application's overall efficiency. Tests conducted on a dataset of 6,000 fruit images revealed that the system can reach an accuracy rate of 87%.

Index Terms **- autoencoder,gateway control model, convolution layer**.

I. INTRODUCTION

In past few years advancements in chip miniaturization and image sensor technology have accelerated significantly. Artificial intelligence algorithm have seen rapid development and broad application across various domains. Technologies have proven especially effective in image processing, where their pattern recognition capabilities align closely with human visual perception. By learning from data, models can extract features, which supports the application for recognizing fruits. Given any fruit quality classification remains a vital yet labor-intensive task in agriculture—typically reliant on human vision and manual inspection. To support farmers and decrease the reliance on manual labor, we have developed an application capable of automatically sorting fruits based on their external quality. Our goal is to deliver an efficient, AI-powered classification system that minimizes hardware costs while maintaining high accuracy.

II. LITERATURE REVIEW:

Krizhevsky pioneered the training of large-scale CNNs using very high-resolution pictures in 2017, Howard et started depthwise(DW) convolution, a highly most efficient convolution operation that decreases both computational cost and the number of parameters. In DW convolution, each input channel—typically three for RGB images—is processed independently with a separate convolution kernel on a two-dimensional plane. Unlike traditional convolution, the actual output tensor size corresponds to those number of filters.

The algorithm YOLO (You Only Look Once) is used for real-time image detection tasks. In 2017, Apte applied YOLO to a mobile application for real-time image classification, enhancing the model's inference speed by modifying the original feature extraction network. The previous work was based on Iandola et al.'s fire module, which reduced parameter count while accelerating image detection. Later, Redmon and Farhadi introduced YOLOv3, a significantly improved version capable of processing 608×608 images at 20 frames per second.

YOLO has also been adopted in security applications. Santad used it to monitor camera footage and track object movement, specifically for luggage-owner association [9]. In the agricultural domain, Yogesh created am important feature database that included fruit characteristics such as color, size, shape,fressness and texture. They segmented images and used an algorithm called

Speeded-Up Robust Features (SURF) algorithm to find out fruit defects. Similarly, Khaing employed a backpropagation-trained CNN for automated fruit classification.

Finally, Chen integrated AI algorithms with microprocessors to automate conveyor gate control for different type of fruits and vegetable segregation system [8]. Their work highlights the growing trend of combining intelligent image processing with embedded hardware for efficient agricultural automation.

III. SYSTEM ARCHITECTURE

A GUI is implemented for demonstrating the real-time position and its classification results. This YOLO algorithm enables efficient fruit detection, even in cases where objects overlap, and supports continuous tracking of individual fruits. A fruit identification model is designed to evaluate the quality of each detected fruit and assign it to the appropriate classification category. An additional model determines whether the fruit meets the required quality standards.

As illustrated in Figure 1, the architecture of this fruit quality detection classification system comprises two supervised learningbased neural network models. Model-A is responsible for object detection and is trained using YOLO algorithm. Model-B is built on a convolutional neural network (CNN) architecture and is implemented for quality recognition. This model learns functional mappings from input data features to quality labels through supervised training. Once validated, the system performs real-time fruit classification and control operations.

Gateway of Control Module :- The module is responsible for controlling the rear and outer gates on the conveyor platform, guiding fruit movement according to the classification output. Neural Network Models : - Two CNN-based models are developed in the system. Model-A leverages the architecture for fruit detection on the conveyor. In contrast, Model-B performs fruit type classification and inspects for surface imperfections to evaluate quality. Bounding Box Optimization:- A comparison and elimination algorithm is implemented to remove redundant bounding boxes generated during the detection phase. This ensures that each fruit is accurately repressented by a single bounding box. Fruit Tracking Algorithm :- This component continuously tracks the fruit along the conveyor platform to maintain identity consistency throughout the classification process. To enhance performance and minimize inference delay, the trained models are optimized and compressed with NVIDIA TensorRT. Figure 2 presents the complete software architecture.

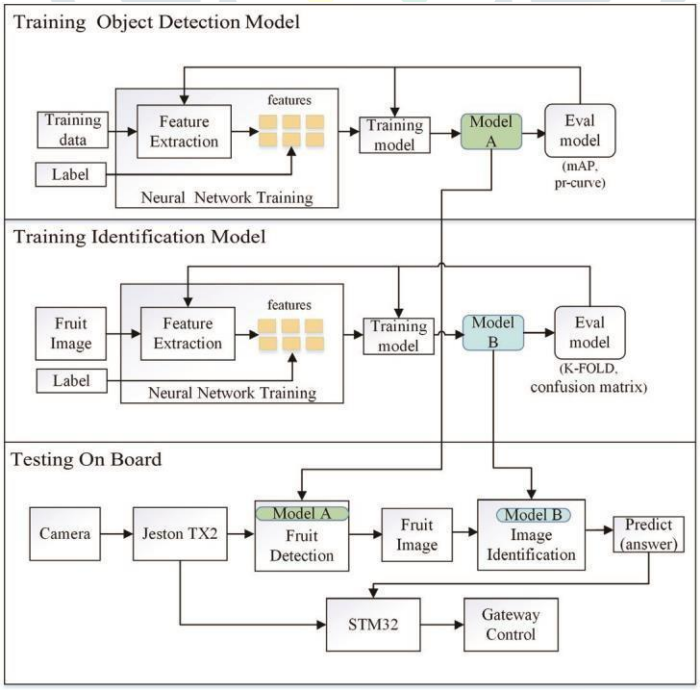

Figure 1. Block Diagram

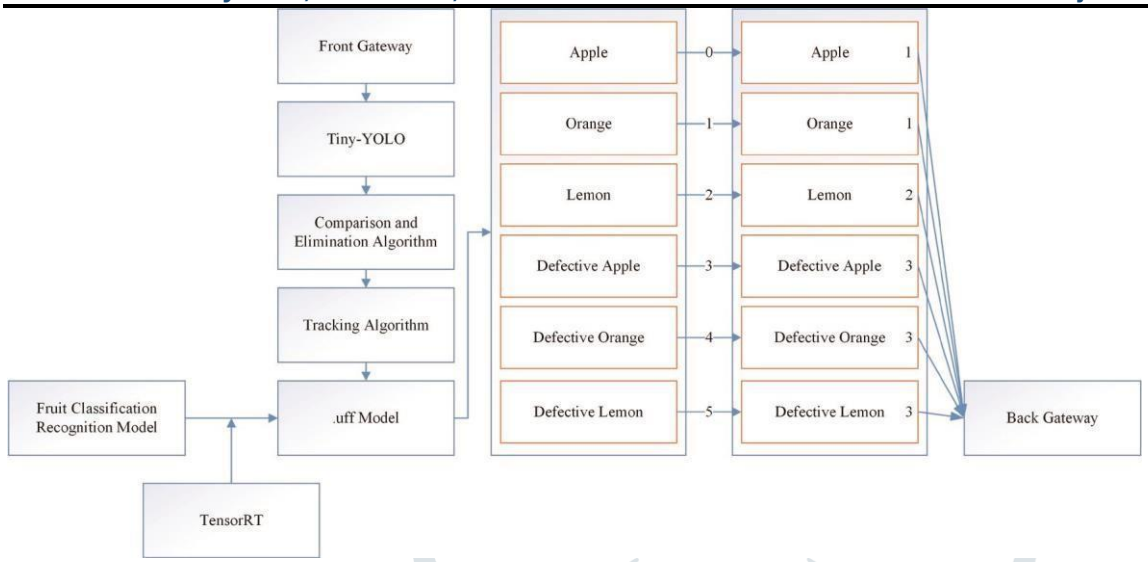

FIGURE 2. OVERVIEW OF COMPONENTS

IV. RESEARCH METHODOLOGY

Model-A: Real-Time Detection of Fruit Objects via Convolutional Neural Network is based on the Tiny-YOLO algorithm, which serves as the foundational framework. Various feature extraction networks are explored and designed to enhance detection performance. The system classifies fruits into two categories: *good* and *defective*. By simplifying the classification into binary categories, the processing time is significantly reduced, aiding in faster convergence of model and accelerating the dataset creation.

Figure 3 presents the structural flow of the architecture used for fruit object detection. The model begins with an input layer that receives the image frames from the conveyor system. It then processes the input through multiple layers, each followed by Batch Normalization (BN) to stabilize and accelerate training. These layers are combined with MaxPooling operations, which progressively reduce the dimensions. The extracted features pass through additional convolutional layers to detect fruit objects at different scales. Finally, the output layer generates bounding boxes and class probabilities, allowing the system to localize and identify fruits in real time.

4.1 Redundancy Removal Algorithm

By analyzing the spatial distance between bounding boxes, it ensures that duplicates are removed, streamlining the process. It's a smart way to decrease errors during object detection. This method guarantees that detections of the same fruit object are reduced to a single, accurate identification, improving performance of the detection process. This technique improves object tracking accuracy by addressing issues caused by fruit defects being partially hidden by undamaged areas. The removal rule of the algorithm ensures that when both defective and non-defective bounding boxes are detected at the same perticular location, the system keeps the box showing the defect and discards the one without it.

Type	Filters	Size	Strides	Padding	Output
Convolution(BN)	16	3x3	1x1	Same	416x416
Maxpooling	16	2x2	2x2	Same	208x208
Convolution(BN)	32	3x3	1x1	Same	208x208
Maxpooling	32	2x2	2x2	Same	104x104
Convolution(BN)	64	3x3	1x1	Same	104x104
Maxpooling	64	2x2	2x2	Same	52x52
Convolution(BN)	128	3x3	1x1	Same	52x52
Maxpooling	128	2x2	2x2	Same	26x26
Convolution(BN)	256	3x3	1x1	Same	26x26
Maxpooling	256	2x2	2x2	Same	13x13
Convolution(BN)	512	3x3	1x1	Same	13x13
Maxpooling	512	2x2	1x1	Same	13x13
Convolution(BN)	1024	3x3	1x1	Same	13x13
Convolution(BN)	256	1x1	1x1	Same	13x13
Convolution(BN)	512	3x3	1x1	Same	13x13
Convolution(BN)	21	1x1	1x1	Same	13x13
Convolution(BN)	128	1x1	1x1	Same	13x13
UpSampling2D	128	Upsampling layer Size=2			26x26
Concatenate	128+256	Merge Layer			26x26
Convolution(BN)	256	3x3	1x1	Same	26x26
Convolution(BN)	21	1x1	1x1	Same	26X26

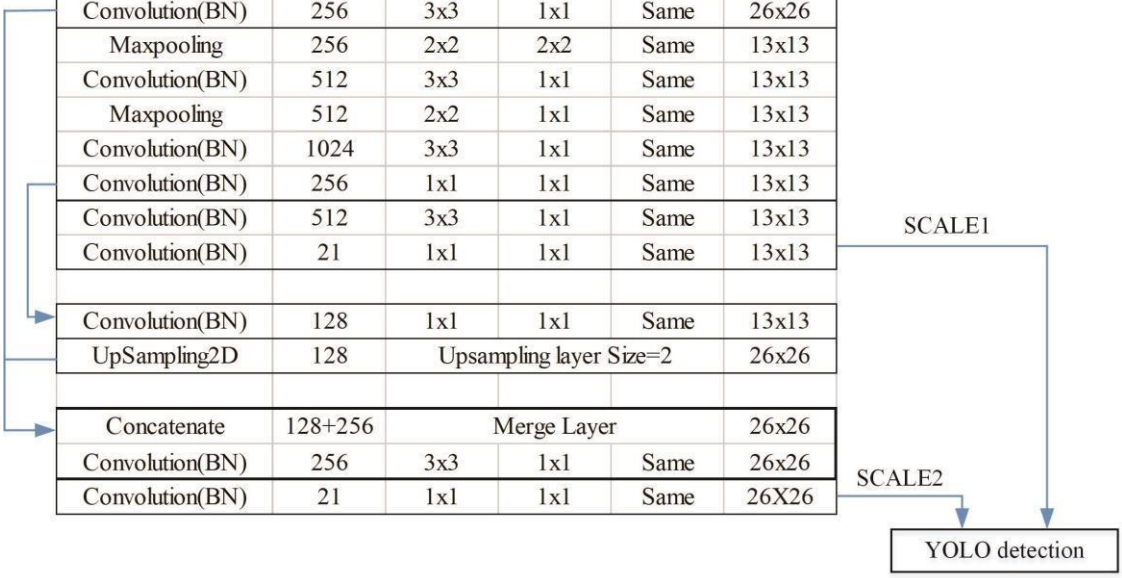

Figure 3. Complete Tiny-YOLO architecture.

4.2 Real-time Object Localization Algorithm

The specified object tracking algorithm is designed for fast computation by utilizing numerical data—specifically the size, height of the detected object, and its confidence score—rather than relying on direct image information. This efficient approach substantially decreases processing time without compromising tracking accuracy. Fruit tracking performs a critical role in improving classification performance, particularly because the fruits tend to rotate or roll as they move along the conveyor platform. By continuously tracking each fruit, the system can observe multiple surfaces, increasing the likelihood of detecting any damaged areas. Before a fruit exits the platform, the system compiles detection statistics by recognizing the classification with the highest occurrence. The final determination direct the fruit for appropriate sorting channel.

4.3 The Categorization and Recognition Model

By utilizing model for extracting the relevant image data, the proposed architecture substantially reduces costs. The model integrates two key architectures: a network, which work together to support the subsequent process. To further optimize efficiency, an interpolation technique is employed to lower the image resolution, thereby speeding up the inference. This method improves performance for the current application but also offers scalability, allowing the method to be adapted for use with a broader range of fruit types in the future.

4.3.1 Feature Extraction Layer and Fully Linked Network

The proposed model architecture contains four convolutional layers, each for deriving data from the input images. After the convolutional layers, a full architecture—highlighted in the green box—performs the final classification. This end-to-end structure allows for the training of all parameters and weights via backpropagation. The result layer consists of nodes, each representing the fruit categories evaluated in the experiment. This architecture enables the model to effectively learn and differentiate between various fruit types during the classification process."

4.3.2 Feature Extraction and Reconstruction Network (Autoencoder)

Figure 4 illustrates the architecture composed of convolution and deconvolution layers, commonly referred to as an autoencoder. The red section represents the encoder, which uses an asymmetric design to gather features from the input image. The green section corresponds to the decoder, which reconstructs the input from the encoded features. The autoencoder architecture is designed to compare the similarity between the reformed output and the original input. If the similarity meets a predefined threshold, the features extracted by the encoder are forwarded to the classification layer. This activity confirms relevant data and thereby improving the performance of the damage detection model.

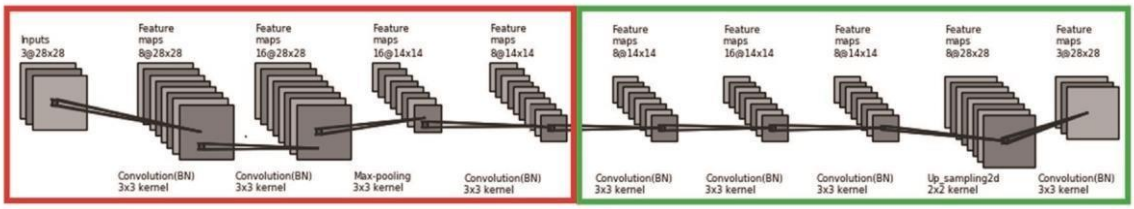

Fig 4. Autoencoder architecture.

4.4 Train and Validate Object Recognition

The system model employs the algorithm for dataset preprocessing. YOLO detects and isolates regions containing fruit within images or video frames. These extracted segments are subsequently classified into 6 types each represented by numeric labels from 0 to 5, as outlined in Table 1. The dataset is partitioned with a 20% allocation for both training and validation. Two identified architectures are evaluated: a neural network and an autoencoder-based network. Both models are trained over 50 epochs. The training process uses the categorical cross-entropy loss function and the Adam optimizer, with an initial lr rate of 0.001. To enhance generalization and reduce computational load, early stopping is employed. Training halts if no improvement in loss is observed for more than 10 consecutive epochs.

4.5 Training model

4.5.1 Layered Structure Using Convolution and Dense Connections

Each layer in the network is designed to allow full training during the learning process, allowing their parameters to be updated throughout the entire training process. Each layer is affected by both forward and backward propagation, enabling continuous learning and adjustment during every training iteration.

4.5.2 Convolution layer and deconvolution layer architecture (autoencoder)

The approach utilizes a sequential, two-phase training process. During the initial phase, an autoencoder is utilized to extract and compress key features by converting the input data into a lower-dimensional form (eigenvalues),then reconstruct an approximation of the original input. Mean squared error is conducted as the loss during model training to quantify reconstruction accuracy by comparing the encoder's input with the decoder's output. To update the model weights effectively, the RMSprop optimizer—recognized for its adaptive learning rate— is used. As illustrated in Figure 8, this phase helps the encoder to learn and retain key features essential for the following classification process.

Table 1
Dataset size in terms of image count

	Apples	Oranges	Lemons	Defective apples	Defective oranges	Defective lemons
The training, verification, and assessment stages	287	405	267	347	291	324
	745	878	639	735	721	749

4.6 k-fold cross-validation

As illustrated in Figure 5, A k-fold cross-validation strategy is employed with k=3k = 3k=3 in this case. The dataset is randomly shuffled and divided into three equal parts. In each iteration, one subset is assigned as the validation set, and the other two subsets are utilized for training. The process is executed three times in total, guaranteeing that each subset functions as the validation data one time. After completing all iterations, the outcomes from each fold are aggregated to compute the overall average accuracy and performance metrics.

The advantage of using k-fold cross-validation lies in its ability to evaluate the model from multiple perspectives, reducing the risk of overfitting and avoiding convergence to a local minimum. It helps the model achieve good generalization across different data partitions.

Figure 6 presents the accuracy results obtained using the architecture, parameters and weights are trained from scratch. The precision of the fully connected model layer architecture starts from zero and gradually increases with training, while the autoencoder-based model begins with an initial accuracy of approximately 80%. In some exceptional cases— illustrated in Figure 6—the accuracy also begins from zero, shows that the corresponding test data differ significantly from the rest. This suggests that the delivery of certain validation samples is not well-aligned with the training data, leading to initial poor performance.

Despite these anomalies, an average accuracy of approximately 87% training epochs increases. Notably, the autoencoder demonstrates an advantage by significantly reducing model training time due to its feature compression capabilities, making it a more efficient solution for real-time or resource-constrained applications.

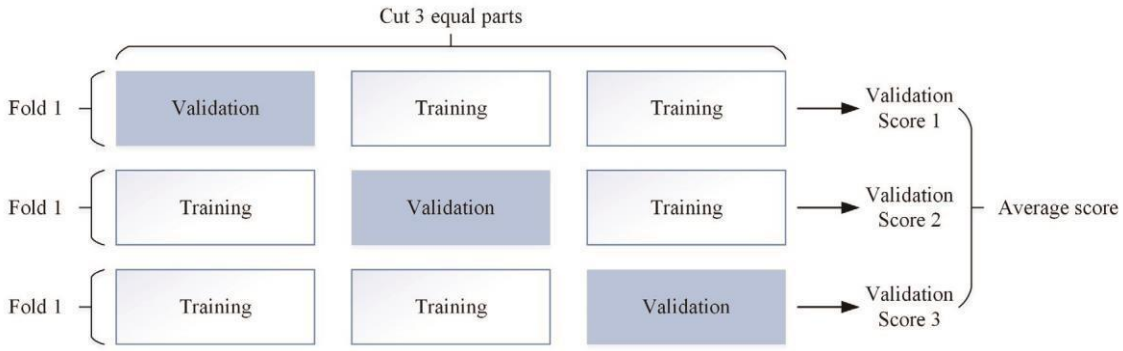

Figure 5: Division in *k*-fold cross-validation system.

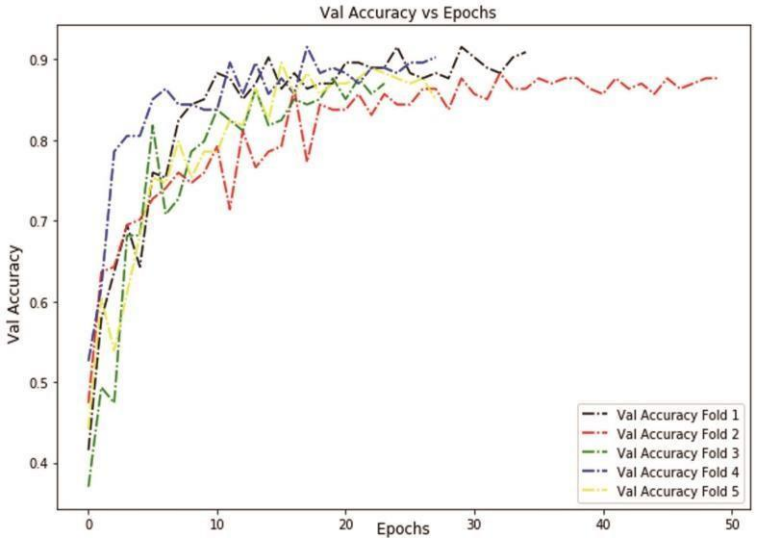

Figure 6. Cross-validation accuracy of connected architecture system.

4.7 Confusion matrix

The confusion matrix provides a valuable tool for evaluating the effectiveness of supervised learning models. It compares data to the model's forecasted output and delivers measuress to assess classification performance. In the case of binary classification, four distinct outcomes are commonly examined, given in Table 2. Tables 3 display the numerical evaluation results for the connected and autoencoder architectures, respectively. These models achieved the performance rate of 88%. For Class 4 (defective oranges), the recall rate was noticeably lower, ranging from 65% to 68%. This represents an approximate 15% decrease compared

to the other classes. The reduced recall for defective oranges can be attributed to the similarities between the images in Dataset 1, leading to misclassification of defective oranges as defective apples.

Table 2
Explaination of confusion matrix.

		Estimation value	
		zero	one
Actual value	zero	True negative (TN)	False positive (FP)
	one	False positive (FP)	True negative (TN)

Table 3

Statistical evaluation of the confusion matrix in the network

	Precision	Recall	F1-score	Support
Class 0	0.83	0.88	0.86	754
Class 1	0.92	0.95	0.94	877
Class 2	0.94	0.95	0.94	639
Class 3	0.83	0.93	0.88	737
Class 4	0.85	0.68	0.76	721
Class 5	0.89	0.89	0.89	748
Avg./Total	0.88	0.88	0.88	4466

V CONCLUSION

We applied the Tiny-YOLO network in our object detection task and analyzed its structural efficiency relative to several competing models. The system was developed using the TensorFlow and Keras frameworks, and images for the dataset were captured with a camera serving as the image sensor.

The algorithm used for recognizing fruits, which incorporated a four-layer convolutional detection model for identifying objects, achieved an impressive accuracy of 88% in classifying fruit quality. The implementation with a graphical user interface, provides a user-friendly interface for operation. Although this study utilized a relatively large model to classify three types of fruit, the strategy may be scaled down for optimized and lightweight design boards. In such cases, a dedicated classification model can be adapted to recognize each fruit type, optimizing the system for more specific application***s.***

VI Acknowledgement

Gratitude is extended to all individuals and institutions whose support and involvement has facilitated this research on FRUIT Quality recognition systems possible.
Sincere thanks are conveyed to **Dr. Suresha D**, Professor and Head of the Department of Computer Science and Engineering, for delivering constructive feedback and leadership and unwavering consistent backing provided during the progress of this research.
Appreciation is also extended to **Dr. Sandeep Bhat,** Associate Professor, for his expert supervision and consistent encouragement. His insightful feedback and technical expertise have significantly enhanced the depth and clarity of the research.
Acknowledgment is given to the broader scientific community, whose foundational work in machine learning, computer vision, and biometric security has enables the development of this study. The continual advancement in these fields provided the necessary tools and inspiration.
Thanks are due to the faculty members and peers whose constructive discussions and thoughtful feedback contributed to refining the methodology and maintaining academic rigor.
Finally, heartfelt appreciation is extended to family and friends for their steadfast encouragement, patience, and moral support during every phase of the research journey.
This endeavor represents a collective effort, and each contribution is sincerely valued.

REFERENCES

[1] A. Krizhevsky, S. Ilya, and E. H. Geoffrey: Commun. ACM **60** (2017) 84. https://doi.org/10.1145/3065386

[2] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam: Comput. Sci. (2017). https://arxiv.org/abs/1704.04861

[3] F. Chollet: Proc. Conf. Computer Vision and Pattern Recognition (IEEE, 2017) 1251. http://doi.org/10.1109/ CVPR.2017.195

[4] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. C. Chen: Proc. Conf. Computer Vision and Pattern Recognition (IEEE, 2018) 4510. http://doi.org/10.1109/CVPR.2018.00474

[5] Y. Chen, H. Fan, B. Xu, Z. Yan, Y. Kalantidis, M. Rohrbach, Y. Shuicheng, and J. Feng: Proc. Int. Conf. Computer Vision (IEEE, 2019) 3435. http://doi.org/10.1109/ICCV.2019.00353

[6] M. Apte, S. Mangat, and P. Sekhar: Stanford University (2017). http://cs231n.stanford.edu/reports/2017/ pdfs/135.pdf

[7] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer: Computer Science (2016). https://arxiv.org/abs/1602.07360

[8] J. Redmon and A. Farhadi: Computer Science (2018). https://arxiv.org/abs/1804.02767

[9] T. Santad, P. Silapasupphakornwong, W. Choensawat, and K. Sookhanaphibarn: Proc. IEEE Global Conf. Consumer Electronics Conf. (IEEE, 2018) 157. https://doi.org/10.1109/GCCE.2018.8574819

[10] Yogesh and A. K. Dubey: Proc. Int. Conf. Reliability, Infocom Technologies and Optimization (IEEE, 2016) 590. https://doi.org/10.1109/ICRITO.2016.7785023