INFORMATICS JOURNALS

Research Article

Journal of Food Sustainability and Security
DOI: 10.18311/jfss/2026/54324 eISSN: xxxx-xxxx | Vol 1(1), March 2026 | pp. 1–9

Design and Development of Automatic Tomato Sorting System using Embedded Electronics and Machine Learning

Ankur Kumar Singh, Akash Sharma, Bhupendra M Ghodki*, Tridib Kumar Goswami

Department of Agricultural and Food Engineering, Indian Institute of Technology Kharagpur – 721302, West Bengal, India; bmg@agfe.iitkgp.ac.in

Abstract

The increasing demand for high-quality tomatoes and price fluctuation have necessitated the adoption of automated sorting systems that ensure accuracy, consistency, and efficiency in post-harvest operations. Thus, this research aimed to design and develop an automatic tomato sorting system that integrates embedded electronics, image processing, and machine learning. A prototype was built using the Arduino Uno microcontroller, a colour sensor, an infrared sensor, and servo motors, with a facilitating image capture and processing. Tomatoes were sorted by colour/ripeness and categorised into three groups: red (ripe), yellow (semi-ripe), and green (unripe) using a convolutional neural network trained on a custom dataset. Image preprocessing techniques, including pixel normalisation and histogram equalisation, were employed to enhance feature extraction. The developed system demonstrated classification accuracy of up to 97.9%, signifying robustness under varied lighting conditions. Besides, the system is suitable for practical implementation in small-scale agricultural operations. This scalable solution promises to reduce manual labor, improve sorting precision, and minimise post-harvest losses, making it a valuable tool for enhancing the tomato supply chain.

Keywords: Arduino Nano UNO, Automation, IR Sensor, Robotics, Tomato

(Article Received on: xx.xx.202x; Revised on: xx.xx.202x; Accepted on: xx.xx.202x)

1. Introduction

The agriculture sector in India provides unswerving employment to more than 50% of the workforce and contributes 16% to the Gross Domestic Product (GDP). The choice of crop cultivation is mainly determined by cropping patterns and physical, economic, and social factors. India is one of the largest producers of tomatoes, with significant export potential. Tomato (*Solanum Lycopersicum)* is a popular horticultural vegetable. Tomatoes and tomato products are one of the most familiar vegetables in our diet. Quantitatively, they are the most consumed non-starchy vegetables and the most significant source of dietary lycopene, a powerful antioxidant with greater bioavailability after cooking and processing (*e.g.*, canning)[1]. Tomatoes are widely used and important vegetables. It is grown for its fruit and is used in varieties of ways for the production of puree pastes, juices and canned fruits or mixed in chilli sources Tomato fruit is found to have high amount of vitamin C. the seed contains 22-29% crude fat, 15-28% crude fibre, 5-10% ash content and 23-34% crude protein according to Standardly in the research by Tsado[2]. As the tomato plays a vital role in day-to-day life, sorting of tomatoes is necessary in evaluating agricultural produce, meeting quality standards, and increasing market value. It is also helpful in planning and packaging. In India, human power in the agricultural sector is widely used. If the sorting and grading are done through manual techniques, the process will be too slow, and sometimes it will be

*Author for correspondence

prone to error. Therefore, an automatic sorting system is proposed to increase efficiency.

A wide range of technologies has been developed or refined over the years for sorting according to colour, density, diameter, size, shape, and weight[3-5]. Zhang *et al.* (2014)[6] developed a machine vision system to automatically sort cherry tomatoes according to maturity. Nine features were extracted from each image. Tomatoes were classified into three categories (unripe, half-ripe, and ripe). Images were captured in the RGB colour space. Bhavana and Reshma (2022)[7] developed a tomato quality evaluation using image processing. The proposed method provides both structural and texture features of the input tomato image. The extracted features are compared by using an Artificial Neural Network (ANN) and a K-means clustering algorithm. Good and spoiled tomatoes can be detected using an edge-detection algorithm. If the image contains more edges, it's considered spoiled; otherwise, it's good. This method is applied only to a single tomato. Mojgan et al. (2019)[8] developed a method for determining the ripeness of tomato fruit juice using image processing and neural network classification.

Current systems often struggle to achieve high accuracy and consistency in sorting due to the inherent variability in tomato size, shape, ripeness, and colour. Moreover, these systems frequently struggle to adapt effectively to different tomato varieties and fluctuating environmental conditions, such as lighting and temperature. Traditional image processing techniques fall short in capturing subtle quality features, which further impacts reliability. Therefore, there is a growing demand for innovative, robust, and adaptable machine learning approaches specifically tailored for tomato sorting applications. Integrating advanced machine learning techniques, such as Convolutional Neural Networks (CNNs) and transfer learning, can potentially improve classification accuracy and operational efficiency. These models can learn complex patterns and generalise across diverse datasets, making them ideal for handling the intricacies of tomato sorting. Additionally, real-time implementation of these systems could significantly reduce human intervention, labour costs, and post-harvest losses. Developing such intelligent systems will enhance the overall quality control process in the tomato supply chain. The basic objectives of this research paper were (i) to develop an algorithm for feature extraction of tomatoes from the captured images. (ii) To train machine learning models for the sorting of tomatoes. (iii) To develop and test an automatic tomato sorter.

2. Materials and Methods

2.1 System Architecture and Overview

The proposed tomato sorting system consists of three integrated subsystems:

- *Sensing Module*: Detects presence and measures the colour of tomatoes using infrared and RGB colour sensors.
- *Control Module:* Processes sensor data, runs classification algorithm, and controls actuation elements via an embedded microcontroller.
- Actuation Module: Physically sorts tomatoes into designated bins based on classification output using servo-driven mechanical components.

Tomatoes are placed on an input platform, which is then detected by an Infrared (IR) sensor. After that, tomatoes are transported to a colour-sensing position where they are classified into ripeness categories using the trained CNN model, and then directed into appropriate collection bins using servo-actuated mechanical sorting elements.

2.2 Hardware Components and Description

2.2.1 Microcontroller Unit

The core control unit is an Arduino Nano Uno-based microcontroller board (16 MHz clock, 2 KB SRAM, 32 KB flash memory). The Arduino Uno is powered either through the USB port or an external 9-12V power supply. It manages signals from all sensors, executes the classification logic, and generates Pulse-Width Modulation (PWM) signals to control servo motors. Communication with the colour sensor was established via the I2C protocol using standard libraries (Wire.h and Adafruit_TCS34725.h).

2.2.2 Colour Sensing Module

The TCS34725 Colour Sensor is an advanced RGB and clear light sensor that delivers highly accurate colour detection, making it ideal for applications such as fruit sorting and quality assessment. It features an integrated

IR-blocking filter that minimises the influence of infrared light, enabling more precise colour measurement across various lighting conditions. The sensor includes four separate Analog-To-Digital Converters (ADCs) for simultaneous detection of red, green, blue, and clear light channels. With a dynamic range of 3,800,000:1 and programmable gain and integration time, it ensures consistent performance across different environments. It communicates via the I2C protocol, making it easy to interface with microcontrollers such as the Arduino Uno. Additionally, it comes with an onboard white LED and MOSFET driver for illuminating objects during detection, ensuring reliable readings even in low-light conditions.

2.2.3 Presence Detection Module

An infrared proximity sensor module was positioned at the input platform to detect the arrival of a tomato. Upon detection, the module triggers the colour sensing sequence and initiates the sorting routine.

2.2.4 Actuation Elements

MG996R servo motors (operating voltage 4.8-6 V, torque 2.5 kg/cm, speed rotates within a range of 0° to 180°) were used for: (i) Rotating the tomato platform towards the colour sensor. (ii) Anotherwas to direct the tomato into the appropriate bin based on colour classification.

The motor's position was controlled by PWM signals, enabling smooth and accurate movement. Its plastic gear construction ensures lightweight operation, suitable for low-load applications such as tomato sorting.

A DIY 2DOF Metal Robot Arm Kit featuring a precision-designed aluminium alloy structure. The kit includes a manipulator claw, servo mounting brackets, and essential accessories, such as nuts, bolts, and ball bearings, enabling seamless assembly of a functional robotic arm. The gripper claw can handle objects weighing up to 75 g, with a maximum gripping width of 54 mm and a total closed length of 90 mm. The complete structure is lightweight, weighing approximately 64 g, and is designed to interface effortlessly with servo motors for smooth articulation. This setup is especially useful in simulating pick-and-place operations within a controlled environment, supporting payloads up to 80 grams when used with an appropriately designed robotic base.

2.2.5 Power Supply and Supporting Components

- *Regulated 5 V DC Power Supply*: Step-down transformer (230 V AC → 12 V AC), full-wave rectifier (bridge circuit), capacitive filter, and linear voltage regulator with input/output filter capacitors.
- *LED Indicators*: LED was used to indicate the active state of sensors, motor functions, and the completion of sorting actions. They provide immediate feedback to the user, making system monitoring easier without the need for external displays.
- *Breadboard and Jumper Wires:* For rapid prototyping and interconnection of components.
- *JHD 162A LCD Display*: 16 × 2 character display for real-time status feedback and debugging information.

2.3 Software Architecture and Embedded Programming

The embedded control software was written in C/C++ using the Arduino IDE. The program structure follows a state-machine approach:

State 1 - Idle: The system waits for IR sensor input.
State 2 - *Detection*: IR sensor signals the presence of a tomato; Arduino initialises colour sensor reading.
State 3 - Colour Measurement: TCS34725 sensor reads RGB values; readings are averaged over multiple samples to reduce noise.
State 4 - Classification: RGB values are passed through the trained CNN model (either pre-trained weights embedded in memory or via serial communication with a connected personal computer (PC)).
State 5 - Actuation: Based on classification output, Arduino sends PWM commands to servo motors to route the tomato to the appropriate bin.
State 6 - Reset: System returns to idle state and awaits the next tomato.

Key libraries used are: (i) Wire.h for I2C communication with TCS34725, (ii)

Adafruit_TCS34725.h for sensor initialisation and data acquisition.

In the embedded CNN interference, two approaches are implemented:

- *On-board Inference*: Pre-trained model weights quantised and stored in Arduino flash memory,

enabling fully autonomous operation (requires model size optimisation).

- *Host-assisted Inference*: Arduino sends RGB values to a connected PC via serial communication (USB/UART); PC run the full CNN model and returns classification output to Arduino.

Arduino Serial Monitor was used during development to observe real-time sensor values, verify correct timing of state transitions, and debug hardware-software integration issues. For circuit validation and microcontroller simulation, Proteus 7 Design Suite is employed to model the hardware schematic and simulate embedded code behaviour prior to physical deployment.

2.4 Tomato Samples and Dataset Preparation

Tomatoes were divided into six ripeness categories. The different ripening stages of each were characterised according to the United States Department of Agriculture.

During model evaluation and quality assessment, one of the most important factors is the use of appropriate datasets. In this study, the proposed solution used image processing and machine learning to determine tomato quality. For the machine learning component, a custom dataset of 678 tomato images was prepared (226 images per ripeness class). Images were obtained from two sources: direct photography with a standard digital camera under controlled indoor lighting, and selected internet sources (augmented samples carefully validated and labelled).

Each tomato instance in the images was annotated and labelled according to ripeness class using a custom Python-based annotation tool. At least two independent observers will ensure accuracy by verifying the ground truth label. The final dataset for CNN training was randomly split into a training set (370 images), a validation set (50 images), and a test set (50 images), for a total of 226 images. Dataset diversity was ensured by including tomatoes with various sizes, under different lighting conditions (soft indoor lighting, bright artificial light, natural daylight), across multiple background conditions (white, grey, and coloured surfaces), and with minor surface imperfections and blemishes on some fruits.

2.5 Image Preprocessing Pipeline

To ensure uniformity and robustness, the following preprocessing steps were applied to all tomato images:

Step 1 - Resizing: All images were resized from their original dimensions to a standardised size of 128 × 128 pixels for efficient training and processing of the Convolutional Neural Network (CNN).

Step 2 - Normalisation: Pixel values were normalised to the range [0, 1] using the Data Generator tool. This improves convergence during neural network training and reduces numerical instability in gradient calculations.

Step 3 - Data Splitting: The dataset was split into 80% for training (182 images) and 20% for validation (44 images). This allowed the model to learn features and patterns in the tomato images, while the validation set provided an unbiased evaluation of its performance during training. These preprocessing steps were essential for preparing the tomato dataset for the CNN model. By standardising the image dimensions, normalising the pixel values, and splitting the data into training and validation sets, the model was able to learn the relevant features and patterns more effectively, leading to improved classification accuracy for the automatic tomato sorting system.

Step 4 - System Process: It classifies tomatoes based on colour maturity, size, and defect degree and sorts them into three levels: premium, first, and second. To check if all system components are functioning normally, click the start button on the user interface. Once the detection is successfu

Step 5 - Feature Engineering (Optional): Handcrafted features were extracted and evaluated alongside deep learning including texture features like Local Binary Patterns (LBP) and Gray-Level Co-occurrence Matrix (GLCM) to capture surface texture,

colour histogram features for distribution of pixel values in R G B channels, statistical features like mean, variance, skewness of RGB channels and morphological features like circularity, solidity and aspect ratio.

2.6 Model Setup and Data Preparation

2.6.1 Dependencies

The following Python libraries and frameworks were used for model development:

- TensorFlow 2X and Keras API for neural network implementation.
- NumPy for numerical computations.
- Pandas for data organisation and manipulation.
- Scikit-learn for pre-processing and feature scaling.
- Matplotlib and Seaborn for visualisation.
- OpenCV for image processing operations.

2.6.2 Data Set Preparation and Training

The training pipeline involved:

- Loading images from the training directory using batch processing.
- Applying data augmentation techniques (rotation, flipping, brightness adjustment) on-the-fly.
- Computing loss and accuracy metrics on both training and validation sets.
- Saving model checkpoints at each epoch for later retrieval of best weights.
- Implementing early stopping to prevent overfitting when validation loss plateaued.

2.6.3 Model Training Parameter Tuning

During model training, to continuously improve detection accuracy, the number of training rounds is adjusted, as are the model's parameters. Different strategies are used to adjust the learning rate while keeping other model parameters constant. Initially, using the linear attenuation approach, a consistent decrease in the learning rate throughout training was noted. This strategy is simple and intuitive and can balance training speed and generalisation ability to a certain extent. However, the learning rate becomes too low during late training, hindering further model optimisation. The cosine attenuation strategy maintains a small learning rate during the late training period, helping avoid overfitting and improving the model's generalisation. Consequently, the learning rate was adjusted using the cosine attenuation strategy to achieve better overall performance.

2.6.4 Performance Matrices

The sensitivity, specificity, and accuracy matrices were used to evaluate model and system performance as discussed in this section.

Sensitivity: Sensitivity is one of the performance evaluation criteria of binary classification tests, and it is also used in artificial intelligence. Sensitivity measures the ratio of positively identified positives, meaning that the classifier can correctly identify the data in classes and calculated using Eq. (1).

$$\text{Sensitivity} = \frac{TP}{TP + FN} \tag{1}$$

where, TP is the correct number of first-class predictions, and FN is the incorrect number of second-class predictions

Specificity: Specificity measures the proportion of negatively identified negatives (for example, the percentage of healthy tomato samples that are correctly identified as not rotten) and calculated using Eq. (2).

$$\text{Specificity} = \frac{TN}{FP + TN} \tag{2}$$

where, TN is a true negative, or healthy tomato samples are properly identified as healthy. The FN is a false negative, or a diseased/unhealthy tomato is incorrectly identified as healthy.

Accuracy: Accuracy is the most intuitive measure of performance and is simply the ratio of the predicted correct observations to the total observations. It may be thought that our model is the best if we are very careful. In general, accuracy is a valuable measure when there is a symmetric data set where the negative, positive, and false positives are approximately the same. Therefore, to evaluate your model's performance, the researchers should consider other parameters. Therefore, the model performance was evaluated using Eq. (3).

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

In a classification task, the accuracy for a class is the number of actual positives divided by the total number of elements labelled positively related to the class (*i.e.*, the sum of the actual positives and False positives are items that are incorrectly labelled as belonging to the class). The recall is defined as the number of true positives divided by the total number of elements that

actually belong to the positive class (*i.e.*, the sum of false positives and false negatives).

2.7 Mechanical Design and System Integration

The mechanical design of the sorting prototype includes an input platform (inclined conveyor or chute (adjustable angle 15-25°) where tomatoes were placed; IR sensor mounted at the end of this platform), colour sensor station (fixed position where tomatoes roll or were held for colour measurement; TCS34725 sensor positioned 5-10 cm above tomato surface with consistent white LED illumination), sorting arm (2DOF robotic gripper arm mounted on a rotating base, capable of reaching three output bins arranged in a semicircular pattern), output bins (three labelled compartments for green, yellow, and red tomatoes, each with capacity for approximately 20-30 fruits during prototype operation), frame structure (Aluminum extrusion or steel framework providing structural support and ensuring stability during operation).

The integration of mechanical, sensing, and control subsystems was validated through iterative prototyping. The Arduino-based controller was programmed to coordinate IR detection, colour measurement, and servo actuation in precise sequential timing to ensure smooth handling and prevent collisions or misplacement. Mechanical tolerances and servo response times were characterised experimentally to ensure robust operation.

3. Results and Discussions

In this section, the results related to the performance of automatic tomato sorters, such as accuracy, determination of colour based on ripeness, and sorting are discussed.

3.1 System Overview

The implemented system for tomato sorting and quality evaluation demonstrated a practical, efficient approach to automating colour-based sorting. The primary objective of this research was to design a reliable embedded system using an Arduino Uno microcontroller, TCS34725 colour sensor, IR sensor, and servo motors for the identification and segregation of tomatoes into predefined categories: green, yellow, and red—based on their ripeness and quality.

3.2 Mechanical Design of Tomato Sorter

The mechanical design of the tomato sorter prototype is a compact and innovative assembly that integrates simple yet effective components to achieve automated sorting. The entire structure was mounted on a flat baseboard, which provides a stable foundation for the components and ensures minimal vibrations during operation. The baseboard was fabricated from a sturdy material to support the weight of the motor, sensors, wiring and other critical parts. The core of the system features a rotating arm mechanism powered by a servo motor, which serves as the primary actuator, directing tomatoes into specific sorting compartments. This servo motor was mounted vertically on the frame and was well-secured with adhesive and mechanical supports to maintain alignment and durability. The arm attached to the motor has a fork-like or shovel-type gripper made of lightweight material, likely aluminium or acrylic, which helps scoop or push the tomatoes based on the sorting signal received.

The sorter was designed with three separate output compartments or trays made from card-like materials that are neatly placed in front of the arm. These trays serve as collection bins for tomatoes sorted according to specific parameters such as size, ripeness, or weight. The placement of the trays was aligned precisely with the arc of the rotating arm, allowing efficient and targeted dropping of tomatoes without overlap. The trays are detachable, making the prototype user-friendly and easy to clean or rearrange based on the needs of the operation. Above the mechanical arm setup, an optical sensor or camera system was positioned to detect tomato characteristics and transmit corresponding control commands to the arm. The entire assembly was mounted on an extended panel, and a green LED indicator displays the system's operational status, as shown in Figure 1.

This model demonstrates the potential for automation in agricultural post-harvest processing. It not only reduces manual labour but also enhances sorting accuracy and efficiency. Such innovations are especially beneficial for small-scale farmers and local processing units. Since tomato quality varies over time, tomatoes of suitable quality tend to have the best shape, colour, and texture. Besides, the tomato's size makes the difference between the first- and second-quality degrees distinguishable. Applying these features together enables the learning model to distinguish each of these

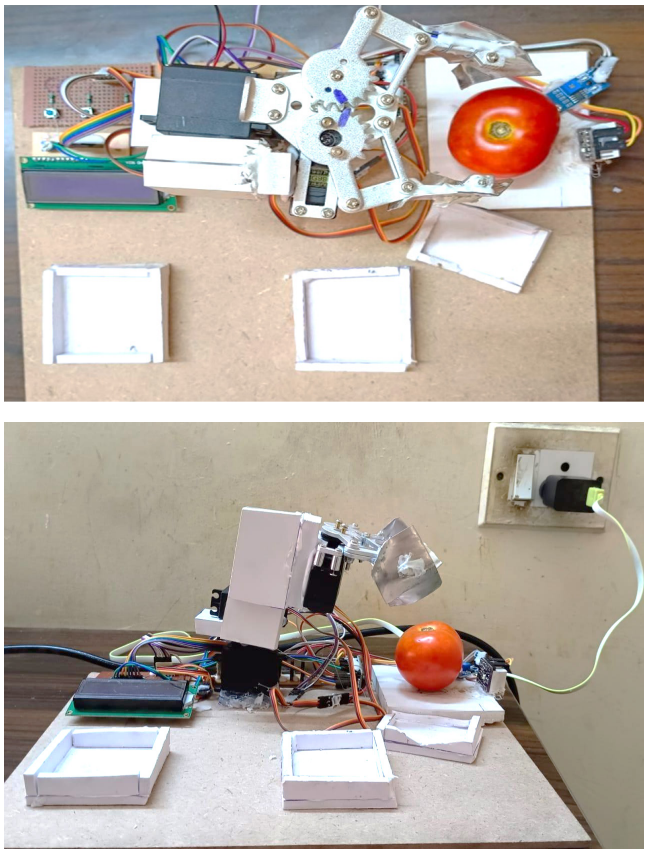


Figure 1. Prototype of the tomato sorting system (top and front views).

quality degrees in later stages. In fact, the background, challenges, and necessities of the study have been analysed. In this section, a method based on machine learning, along with an image processing technique, is presented to improve the quality recognition of tomatoes, and the proposed method was examined in terms of implementation and evaluation. The proposed steps were then evaluated step by step.

3.3 Tomato Classification Analyses

Agricultural products, such as ripe tomatoes, may be damaged during the post-ripening period. Based on the characteristics of colour, a comprehensive overview of the performance metrics of a machine learning model across 10 epochs is shown in Table 1. Each row represents a specific epoch and details the model's training accuracy, validation loss, validation accuracy, and training loss. The critical observations are discussed further.

Epoch and Metrics: The model's training over 10 epochs shows a strong progression in both accuracy and loss, indicating effective learning. Initially, the training accuracy is 0.51 with a relatively high loss of 0.69, and the validation accuracy is 0.82, suggesting good generalisation from the beginning. As the epochs progress, the training accuracy consistently increases, reaching 1.0 by epoch 7, while the training loss significantly drops to 0.05 and even lower in later epochs. Similarly, validation accuracy peaks at 0.97 during epochs 6 and 9 with a corresponding minimal validation loss of 0.009 and 0.01, respectively.

Table 1. Training and validation accuracy and loss across epochs for the sorting model

Epochs	Accuracy	Validation Loss	Validation Accuracy	Loss
Epoch1	0.51	0.69	0.43	0.82
Epoch2	0.49	0.59	0.75	0.66
Epoch3	0.81	0.42	0.86	0.44
Epoch4	0.95	0.21	0.86	0.18
Epoch5	0.98	0.17	0.90	0.05
Epoch6	0.98	0.06	0.97	0.05
Epoch7	1	0.05	0.97	0.009
Epoch8	1	0.42	0.86	0.003
Epoch9	0.99	0.05	0.97	0.01
Epoch10	0.98	0.12	0.93	0.02

*Training versus Validation Accuracy: T*raining and validation accuracy of the tomato sorting model over 10 epochs are represented in Table 1 and Figure 2. Initially, both accuracies increase steadily, indicating effective learning during the early stages. The training accuracy continues to improve, approaching near-perfect accuracy by epoch 4, and then stabilises. In contrast, the validation accuracy also rises but shows fluctuations after epoch 6, suggesting potential overfitting. The divergence between training and validation curves in later epochs implies that the model may be learning patterns specific to the training data. Overall, the model performs well, but fine-tuning may be needed to improve generalisation. This fluctuation in validation accuracy could be addressed by techniques such as dropout, early stopping, or data augmentation. Despite the variance, the high validation accuracy indicates the model's strong potential for reliable tomato sorting based on colour.

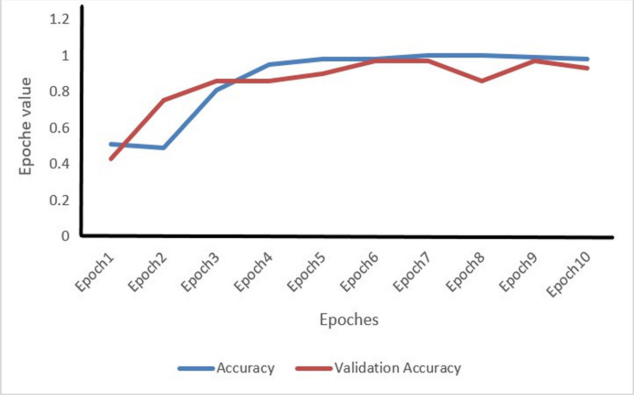


Figure 2. Training versus validation accuracy.

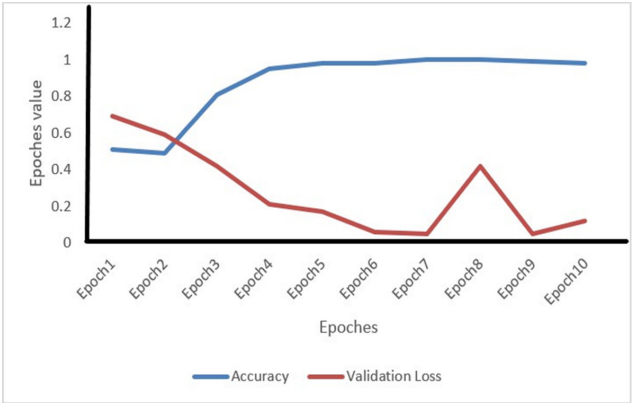


Figure 3. Training versus validation loss.

Training versus Validation loss: Training and validation loss curves of the tomato sorting model over 10 epochs are shown in Figure 3. Initially, both training and validation loss decrease sharply, indicating effective learning and a good fit to the data. By the fourth epoch, the training loss becomes very low and stabilises near zero, reflecting that the model is fitting the training data extremely well. The validation loss also decreases steadily until epoch 6, but then spikes at epoch 7, suggesting temporary overfitting or a batch of complex validation samples. After this spike, the validation loss again drops and remains relatively low, though with slight fluctuation. This overall trend indicates that the model is learning effectively but may be slightly sensitive to certain validation inputs. The consistent decrease in training loss, coupled with low validation loss for most epochs, demonstrates that the model is generally reliable. However, the spike suggests the need for regularisation techniques like dropout or data augmentation to maintain stability. Despite this, the final validation loss remains low, indicating strong generalisation for tomato sorting based on colour features.

Overall Performance: The model shows robust learning and generalisation capabilities across the epochs. Consistently low validation loss and high validation accuracy indicate effective model training.

Accuracy, validation accuracy, training loss, and validation loss across 10 epochs for the tomato sorting model are illustrated in Figure 4. The training accuracy (blue) steadily increases and reaches near 1 by the 7th epoch, indicating that the model is learning the training data well. The validation accuracy (green) rises but shows some fluctuations, peaking at epoch 7 and then declining, suggesting overfitting or instability. Both training and validation losses (purple and red) initially decrease but spike at epoch 7, suggesting challenges with generalisation. Overall, the model performs well, though the validation loss spike indicates the need for further tuning to enhance generalisation. This fluctuation in validation accuracy and loss could be due to varying complexities in the validation dataset or slight overfitting after several epochs. The near-zero training loss from epoch 6 onwards demonstrates the model's high confidence in class predictions. However, the validation loss spike at epoch 8 may reflect sensitivity to unseen data, which can be mitigated by techniques like regularisation or data augmentation. Despite these fluctuations, the model maintains high overall accuracy, making it suitable for real-time colour-based tomato sorting.

3.4 System Robustness, Limitations, and Applications

An important aspect noted during the experimentation was the consistency in colour detection even when the tomatoes slightly varied in size. This robustness highlights the adaptability of the sensor in real-world scenarios where uniformity is not always guaranteed. However, challenges were encountered under low-light or overly bright conditions, leading to occasional misclassifications. This suggests that environmental light calibration or the incorporation of an enclosed sensor chamber could improve accuracy further.

The system's compactness, low power consumption, and real-time response make it highly suitable for small-scale sorting applications, particularly in agricultural settings or local markets. Additionally, the use of open-source components like Arduino enhances the scalability and adaptability of the system for further

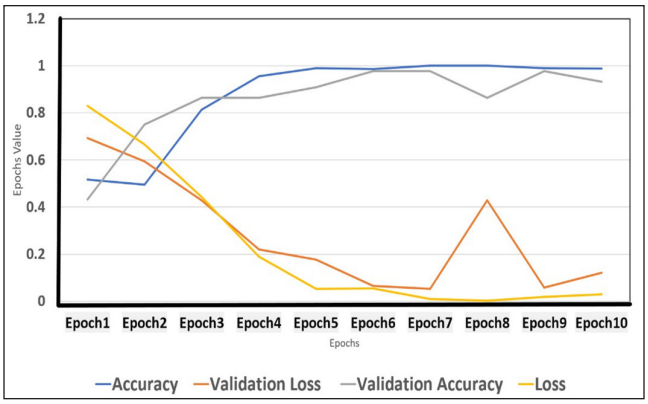


Figure 4. Comparison of accuracy, validation loss, validation accuracy, and loss.

developments, such as integrating size detection or defect analysis.

4. Conclusion

The research successfully demonstrated the development of an automated sorting system for tomato ripeness and quality evaluation, leveraging open-source hardware and advanced image processing techniques. By integrating the TCS34725 colour sensor with a CNN-based classifier, the system achieved over 97.9% classification accuracy, effectively sorting tomatoes into red, yellow, and green categories. The successful implementation of this prototype highlights its potential for scaling in small- to medium-sized agricultural enterprises. It lays a foundation for further enhancements such as size and defect detection, improved lighting calibration, and mobile or integration, contributing to the broader vision of smart and sustainable farming.

5. Acknowledgements

The authors thank the Indian Institute of Technology Kharagpur for all support.

6. References

1. Burton-Freeman B, Reimers K. Tomato consumption and health: Emerging benefits. Am J Lifestyle Med. 2010; 5(2):182-191. https://doi.org/10.1177/1559827610387488
2. Tsado EK. The best source of compost for tomato production: A study of tomato production in Niger state, Nigeria. Global J Agric Res. 2015; *3*(4):23-33.
3. Adamu AA, Shehu A. Development of an automatic tomato sorting machine based on a colour sensor. Int J Recent Eng Res Develop. 2018; *3*(11):1-7.
4. Arjenaki OO, Moghaddam PA, Motlagh AM. Online tomato sorting based on shape, maturity, size, and surface defects using machine vision. Turkish J Agri Forest. 2013; 37(1):62-68. https://doi.org/10.3906/tar-1201-10
5. Sari MI, Fajar R, Gunawan T, Handayani R. The use of image processing and sensor in tomato sorting machine by colour, size, and weight. JOIV: Int J Inform Visual. 2022; 6(1-2):244-249. http://dx.doi.org/10.30630/joiv.6.1-2.944
6. Zhang Y, Wang S, Ji G, Phillips P. Fruit classification using computer vision and feedforward neural network. J Food Eng. 2014; 143:167-177. https://doi.org/10.1016/j.jfoodeng.2014.07.001
7. Bhavana, K. V., & Reshma, K. J. (2016). Tomato quality evaluation using image processing. International Journal of Engineering Research in Computer Science and Engineering (IJERCSE), 3(5), 116–120.
8. Mojgan, E., Dadios, E. P., Bandala, A. A., & Vicerra, R. R. P. (2019). Size classification of tomato fruit using thresholding, machine learning, and deep learning techniques. Agrivita Journal of Agricultural Science, 41(3), 586–596. https://doi.org/10.17503/agrivita.v41i3.2435

Author contribution

Ankur Kumar Singh: Methodology, software, validation, formal analysis, investigation, data curation, visualisation, writing—original draft preparation. **Akash Sharma:** writing—original draft preparation, formal analysis. **Bhupendra M Ghodki:** Conceptualisation, methodology, resources, writing—review and editing, visualisation, supervision. **Tridib Kumar Goswami:** Methodology, resources, writing, review and editing.

Access this journal online	
Website: https://www.informaticsjournals.co.in/index.php/jfss/	Quick Response Code