Automated Vegetable Classification for E-Commerce Applications: Enhancing Online Grocery Shopping

Dr. N. Baskar[1], V. Srija[2], P. Bhavana[2], S. Sandhya[2]

[1]Professor, [2]UG Student, [1,2]Department of Computer Science Engineering
[1,2]Malla Reddy Engineering College for Women, Maisammaguda, Dhulapally, Kompally, Secunderabad-500100, Telangana, India

ABSTRACT

The convenience of online shopping has revolutionized the way consumers procure goods, including groceries. However, in the domain of fresh produce, customers often face challenges in accurately visualizing and selecting items, as they rely heavily on visual cues such as color, size, and shape. Automated vegetable classification aims to address this issue by utilizing technology to assist customers in making informed choices. Conventional e-commerce platforms often rely on manual image tagging and categorization. This process is time-consuming and can lead to inconsistencies due to human subjectivity. Moreover, it doesn't scale well as the variety of vegetables and customer demand increases. The primary challenge in this context is to develop a system that can accurately and swiftly classify vegetables based on visual attributes. This involves training a model to recognize and differentiate between various types of vegetables, taking into account factors like color, size, shape, and texture. Therefore,the online grocery shopping continues to grow in popularity, providing an efficient and user-friendly experience is crucial for e-commerce platforms. Automating the process of vegetable classification can enhance the accuracy and speed at which customers can select their produce, reducing the likelihood of mismatches between expectations and delivered items. This, in turn, boosts customer satisfaction and confidence in online grocery shopping. The project, "Automated Vegetable Classification for E-Commerce Applications: Enhancing Online Grocery Shopping," aims to revolutionize the online grocery shopping experience by employing advanced computer vision and machine learning techniques. By leveraging large datasets of annotated vegetable images, this research endeavors to train a model capable of accurately classifying vegetables in real-time. Through the integration of state-of-the-art algorithms, the system will provide customers with instant and precise visual cues, empowering them to confidently select their produce online. This advancement holds the potential to significantly enhance the efficiency and satisfaction of online grocery shopping, further establishing e-commerce platforms as a reliable source for fresh produce.

Keywords: Enhancing Online Grocessary Shopping, Automated Vegetable classification, E-Commerce Applications.

1. INTRODUCTION

Over the past decade, the landscape of retail has undergone a radical transformation, primarily propelled by the burgeoning trend of online shopping. This paradigm shift in consumer behavior has not spared the domain of grocery shopping, an essential aspect of daily life. The convenience and accessibility offered by e-commerce platforms have redefined the way consumers acquire goods, presenting an unprecedented opportunity for innovation. One critical aspect of this transition, however, has remained a challenge—the selection of fresh produce.

In the realm of online grocery shopping, customers frequently encounter hurdles when attempting to accurately visualize and select fresh vegetables. Unlike non-perishable items that can be precisely described through standardized attributes, the selection of produce relies heavily on nuanced visual cues. These cues, encompassing factors such as color, size, shape, and texture, are crucial in ensuring that customers receive the quality and variety of vegetables they desire. Recognizing the limitations of conventional e-commerce platforms in this regard, there is a pressing need for innovative solutions to enhance the user experience and bridge the gap between the virtual and physical realms of grocery shopping.

One prevailing issue in the current landscape is the reliance on manual image tagging and categorization for vegetable listings. This labor-intensive process not only consumes valuable time but is also susceptible to inconsistencies arising from human subjectivity. As the diversity of available vegetables expands and customer demand continues to surge, the manual approach becomes increasingly untenable. To overcome these challenges, the key imperative is to develop a system that can swiftly and accurately classify vegetables based on their visual attributes.

This initiative sets out to revolutionize the online grocery shopping experience by seamlessly integrating advanced computer vision and machine learning techniques. At its core, the project seeks to address the fundamental challenge of training a model capable of recognizing and differentiating between various types of vegetables in real-time.

The significance of this endeavor is underscored by the rapid growth in popularity of online grocery shopping. As more consumers embrace the digital marketplace for their daily needs, ensuring an efficient and user-friendly experience becomes paramount for e-commerce platforms. The automated vegetable classification system envisioned by this project holds the promise of not only streamlining the selection process but also mitigating the risks of mismatches between customer expectations and the delivered items.

A critical aspect of this project lies in its commitment to leveraging large datasets of annotated vegetable images. These datasets serve as the foundation for training the machine learning model, providing it with the diverse and extensive visual information needed to accurately classify vegetables. The incorporation of state-of-the-art algorithms further distinguishes this project, ensuring that the system operates at the cutting edge of technological capabilities.

2. Literature Survey

Automatic vegetable classification is an intriguing challenge in the growth of fruit and retailing industrial chain since it is helpful for the fruit producers and supermarkets to discover various fruits and their condition from the containers or stock with a view to improvising manufacturing effectiveness and revenue of the business [1]. Thus, intelligent systems making use of machine learning (ML) approaches and computer vision (CV) have been applied to fruit defect recognition, ripeness grading, and classification in the last decade [2]. In automated vegetable classification, two main methods, one conventional CV-related methodologies and the other one deep learning (DL)-related methodologies, were investigated. The conventional CV-oriented methodologies initially derive the low-level features, after which they execute image classification through the conventional ML approaches, while the DL-related techniques derive the features efficiently and execute an endwise image classification [3]. In the conventional image processing and CV approaches, imagery features, such as shape, texture, and color, were utilized as input unit for vegetable classification.

Previously, fruit processing and choosing depended on artificial techniques, leading to a huge volume of waste of labor [4]. Nonetheless, the above-mentioned techniques require costly devices (various kinds of sensors) and professional operators, and their comprehensive preciseness is typically less than 85% [5]. With the speedy advancement of 4G communication and extensive familiarity with several mobile Internet gadgets, individuals have created a large number of videos, sounds, images, and other data, and image identification technology has slowly matured [6].

Image-related fruit recognition has gained the interest of authors because of its inexpensive gadgets and extraordinary performances [7]. At the same time, it is needed to design automated tools capable of handling unplanned scenarios such as accidental mixing of fresh products, fruit placement in unusual packaging, different lighting conditions or spider webs on the lens, etc. Such situations may also cause uncertainty in the model results. The intelligent recognition of fruit might be utilized not only from the picking stages of the prior fruit but also in the processing and picking phase in the next stage [8]. Fruit identification technology depending on DL could substantially enhance the execution of fruit identification and comprises a positive impact on fostering the advancement of smart agriculture. In comparison with artificial features and conventional ML combination techniques, DL may derive features automatically, and contains superior outcomes that slowly emerged as the general methodology of smart recognition [9]. Particularly, convolutional neural network (CNN) is one of the vital DL models utilized for image processing. It is a type of artificial neural network (ANN) which utilizes convolution operation in at least one of the layers. Recently, CNNs have received significant attention on the image classification process. Specifically, in the agricultural sector, CNN-based approaches have been utilized for vegetable classification and fruit detection [10].

In [11], the authors suggest an effective structure for vegetable classification with the help of DL. Most importantly, the structure depends on two distinct DL architectures. One is a proposed light model of six CNN layers, and the other is a fine-tuned visual geometry group-16 pretrained DL method. Rojas-Aranda et al. [12] provide an image classification technique, based on lightweight CNN, for the purpose of fastening the checking procedure in the shops. A novel images dataset has presented three types of fruits, without or with plastic bags.

3. PROPOSED METHODOLOGY

Overview

Figure 1 shows the proposed system model. The detailed operation procedure illustrated as follows: **Dataset:** Begin by curating a dataset with images of vegetables corresponding to the 'vegetables' list, encompassing 'Tomato' to 'Bitter_Gourd.' Ensure the dataset is comprehensive and diverse, showcasing various angles, lighting conditions, and backgrounds for each vegetable.

Dataset (Images) Preprocessing: Prepare the dataset by resizing images to a uniform dimension, normalizing pixel values, and applying augmentation techniques like rotation or flipping to enhance model generalization. These preprocessing steps contribute to a robust and well-conditioned dataset for subsequent model training.

Train-Test Split (0.2): Split the preprocessed dataset into training and testing subsets, allocating 80% for training the Convolutional Neural Network (CNN) and 20% for evaluating its performance. This division ensures the model's ability to generalize to unseen data.

CNN Model Training: Implement a CNN architecture for image classification, defining layers, filters, and activation functions. Train the model using the training dataset, employing optimization algorithms like Stochastic Gradient Descent (SGD) or Adam to minimize the classification error.

Performance Estimation: Evaluate the model's performance using the testing dataset, computing key metrics such as accuracy, precision, recall, and F1 score. These metrics provide a comprehensive understanding of the model's classification capabilities and potential areas for improvement.

Test Vegetable Image: Acquire a new vegetable image for model testing, ensuring it represents a real-world scenario and differs from the training dataset to assess the model's generalization.

Preprocessing: Apply the same preprocessing steps to the test image as done during the initial dataset preprocessing. This step ensures consistency and compatibility between the test image and the training data.

Model Prediction: Utilize the trained CNN model to predict the class (vegetable type) of the preprocessed test image. Extract the model's prediction probabilities for each class to gauge the confidence level of the classification.

Output: Display the model's prediction output, revealing both the predicted vegetable class and the associated confidence level. This information offers insights into the model's certainty regarding its classification.

Language-to-English Conversion Add-on: Implement a language-to-English conversion add-on to facilitate user interaction and interpretation of results, enhancing the system's usability and accessibility for a broader audience. This addition ensures a seamless and user-friendly experience in communicating with the model.

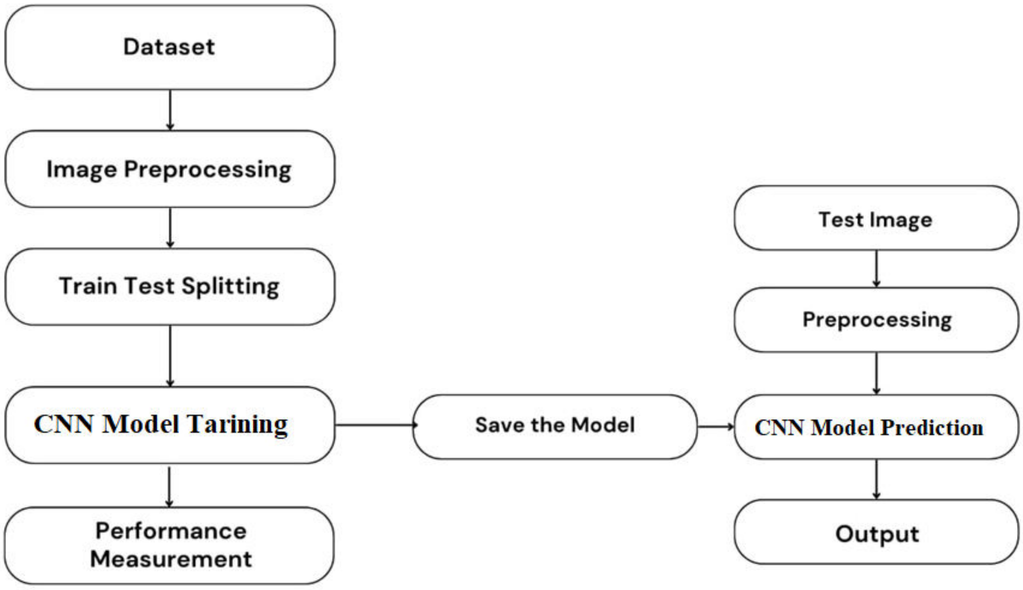

Figure.1: Proposed system model.

Data preprocessing

Image preprocessing is a critical step in computer vision and image analysis tasks. It involves a series of operations to prepare raw images for further processing by algorithms or neural networks. Here's an explanation of each step in image preprocessing:

Step 0. Image Read: The first step in image preprocessing is reading the raw image from a source, typically a file on disk. Images can be in various formats, such as JPEG, PNG, BMP, or others. Image reading is performed using libraries or functions specific to the chosen programming environment or framework. The result of this step is a digital representation of the image that can be manipulated programmatically.

1. Image Resize: Image resize is a common preprocessing step, especially when working with machine learning models or deep neural networks. It involves changing the dimensions (width and height) of the image. Resizing can be necessary for several reasons:

- Ensuring uniform input size: Many machine learning models, especially convolutional neural networks (CNNs), require input images to have the same dimensions. Resizing allows you to standardize input sizes.
- Reducing computational complexity: Smaller images require fewer computations, which can be beneficial for faster training and inference.
- Managing memory constraints: In some cases, images need to be resized to fit within available memory constraints.

When resizing, it's essential to maintain the aspect ratio to prevent image distortion. Typically, libraries like OpenCV or Pillow provide convenient functions for resizing images.

2. Image to Array: In this step, the image is converted into a numerical representation in the form of a multidimensional array or tensor. Each pixel in the image corresponds to a value in the array. The array is usually structured with dimensions representing height, width, and color channels (if applicable).

For grayscale images, the array is 2D, with each element representing the intensity of a pixel. For color images, it's a 3D or 4D array, with dimensions for height, width, color channels (e.g., Red, Green, Blue), and potentially batch size (if processing multiple images simultaneously).

The conversion from an image to an array allows for numerical manipulation and analysis, making it compatible with various data processing libraries and deep learning frameworks like NumPy or TensorFlow.

3. Image to Float32: Most machine learning and computer vision algorithms expect input data to be in a specific data type, often 32-bit floating-point numbers (float32). Converting the image array to float32 ensures that the pixel values can represent a wide range of intensities between 0.0 (black) and 1.0 (white) or sometimes between -1.0 and 1.0, depending on the specific normalization used.

This step is essential for maintaining consistency in data types and enabling compatibility with various machine learning frameworks and libraries. It's typically performed by dividing the pixel values by the maximum intensity value (e.g., 255 for an 8-bit image) to scale them to the [0.0, 1.0] range.

4. Image to Binary: Image binarization is a process of converting a grayscale image into a binary image, where each pixel is represented by either 0 (black) or 1 (white) based on a specified threshold. Binarization is commonly used for tasks like image segmentation, where you want to separate objects from the background.

The process involves setting a threshold value, and then for each pixel in the grayscale image, if the pixel value is greater than or equal to the threshold, it is set to 1; otherwise, it is set to 0.

Binarization simplifies the image and reduces it to essential information, which can be particularly useful in applications like character recognition or object tracking, where you need to isolate regions of interest.

Dataset Splitting

In machine learning data pre-processing, we divide our dataset into a training set and test set. This is one of the crucial steps of data pre-processing as by doing this, we can enhance the performance of our machine learning model. Suppose if we have given training to our machine learning model by a dataset and we test it by a completely different dataset. Then, it will create difficulties for our model to understand the correlations between the models. If we train our model very well and its training accuracy is also very high, but we provide a new dataset to it, then it will decrease the performance. So we always try to make a machine learning model which performs well with the training set and also with the test dataset. Here, we can define these datasets as:

Training Set: A subset of dataset to train the machine learning model, and we already know the output.

Test set: A subset of dataset to test the machine learning model, and by using the test set, model predicts the output.

4. RESULTS AND DISCUSSIONS

Results and description:

The figure 2 shows examples or samples from different classes within the dataset. It provides a visual representation of the variety of classes present in the dataset used for the research work.

Figure. 2: Sample classes presented in dataset.

The figure 3 represents the user-interface screen used for conducting the research work. It has options, tools to the tasks involved in the research.

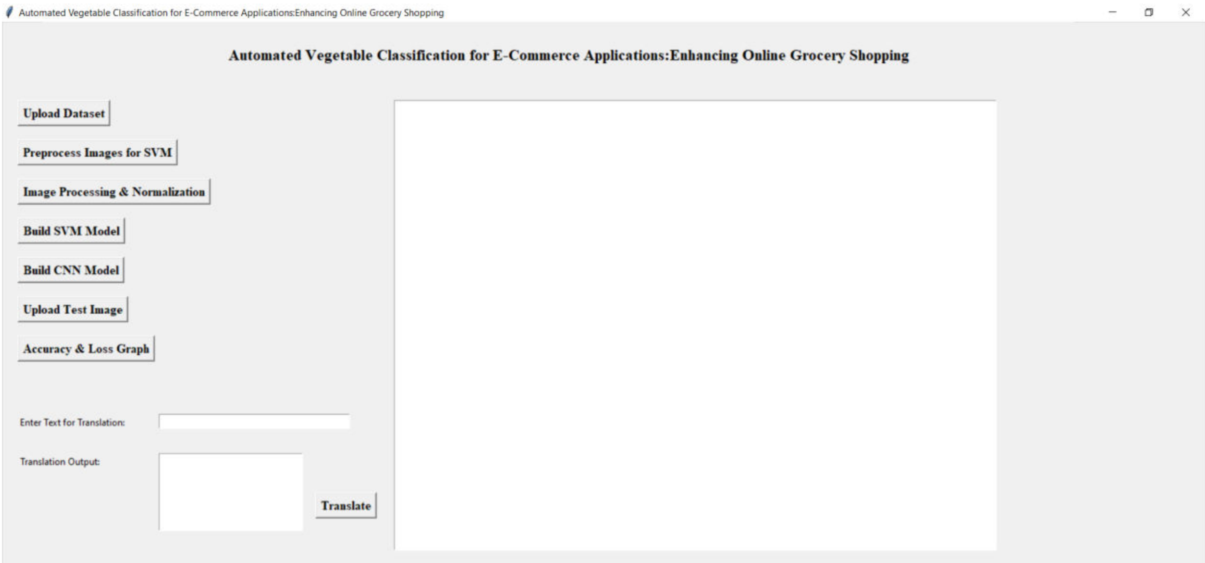

Figure 3. user-interface screen of research work.

The figure 4 displays yet another sample test image predicted as belonging to the class "pumpkin." It illustrates the model's ability to classify different classes.

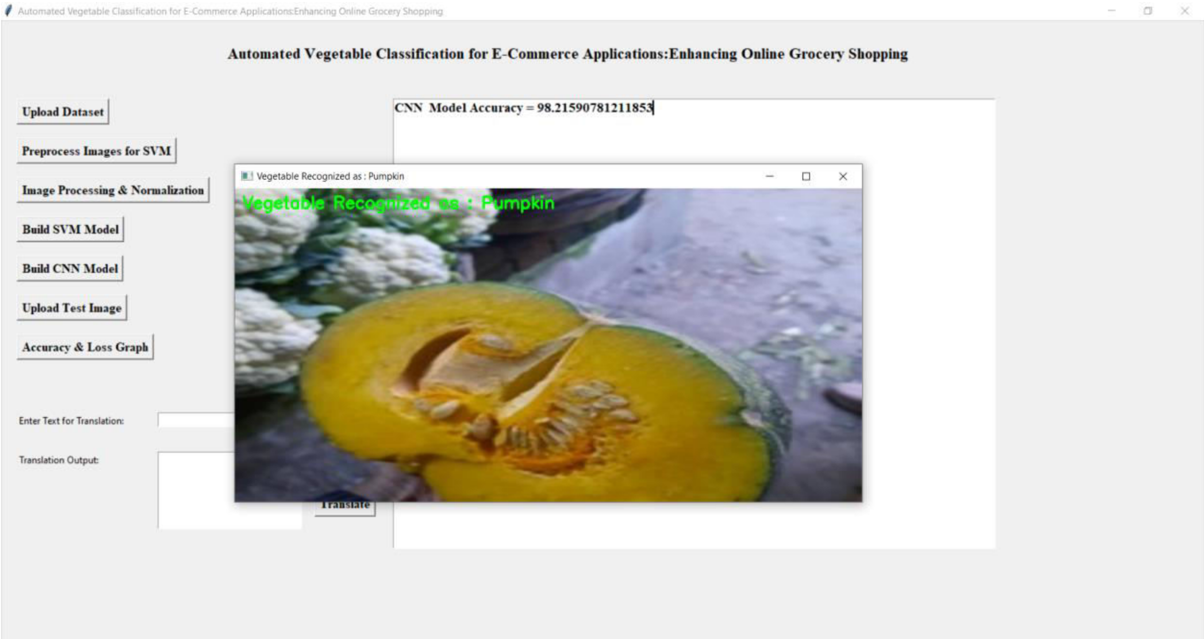

Figure. 4: Sample test image predicted as pumpkin.

The figure likely shows a graph presenting the accuracy and loss of the proposed model during training. It helps visualize the model's learning process over epochs.

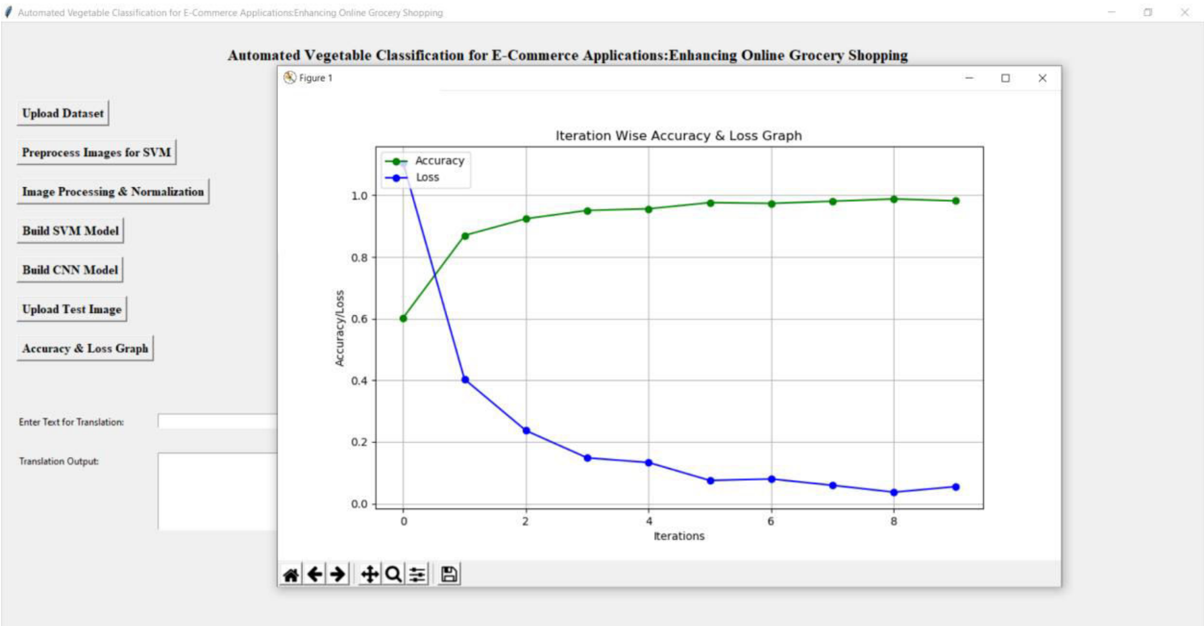

Figure 5. Accuracy and loss graph of proposed model.

The figure represent a scenario where an unknown language is input into the system. It could be a part of language processing or translation tasks.

The figure below shows the outcome of converting the unknown language into English. It demonstrates the effectiveness of the language conversion or translation process in the research work

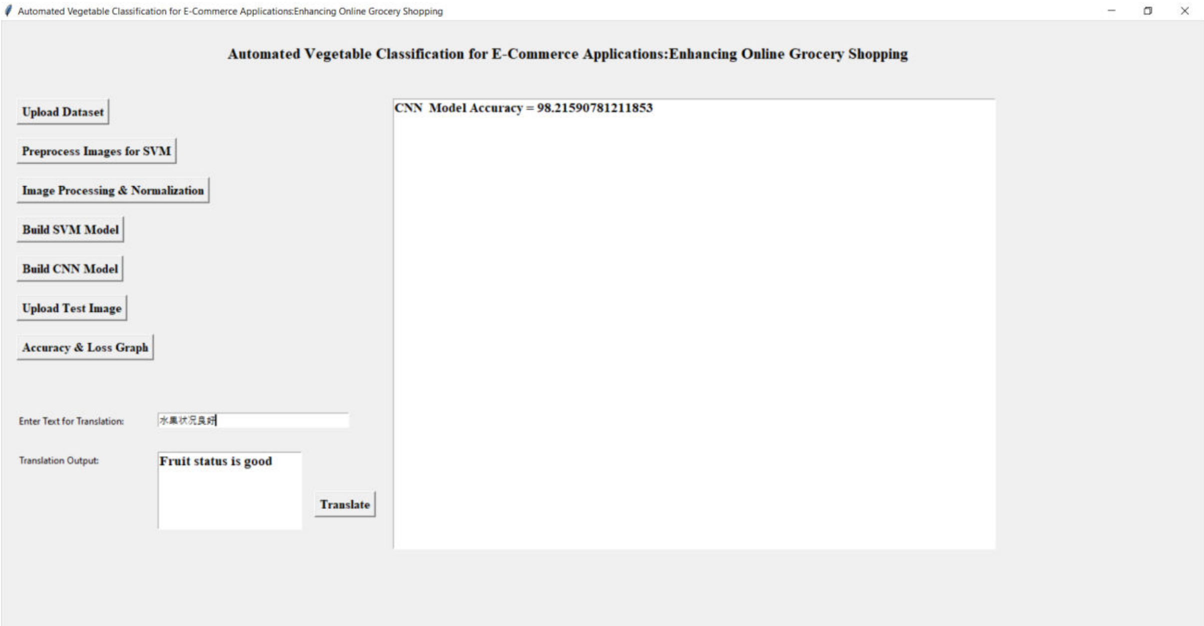

Figure 6. Unknown language to English language converted outcome.

5. Conclusion and Future Scope

In conclusion, the detailed operational procedure outlined above represents a comprehensive workflow for building and evaluating a Convolutional Neural Network (CNN) model for vegetable image classification. The process begins with dataset curation, emphasizing diversity and completeness in capturing various aspects of each vegetable. Subsequent preprocessing steps ensure the dataset's readiness for model training, including resizing, normalization, and augmentation. The train-test split facilitates robust evaluation, with 80% of the data dedicated to training and 20% for assessing the model's performance. The CNN model is then constructed and trained, leveraging optimization techniques to minimize classification error. Performance metrics offer a nuanced understanding of the model's effectiveness, guiding potential refinements. Testing the model with a new vegetable image validates its generalization capabilities, and the language-to-English conversion add-on enhances user interaction. So, The detailed procedure ensures a systematic and thorough approach to developing a reliable vegetable classification system, balancing model intricacies with practical considerations.

REFERENCES

[1] Altaheri, H.; Alsulaiman, M.; Muhammad, G. Date fruit classification for robotic harvesting in a natural environment using deep learning. IEEE Access 2019, 7, 117115–117133.

[2] Chen, X.; Zhou, G.; Chen, A.; Pu, L.; Chen, W. The fruit classification algorithm based on the multi-optimization convolutional neural network. Multimed. Tools Appl. 2021, 80, 11313–11330.

[3] Khan, R.; Debnath, R. Multi class fruit classification using efficient object detection and recognition techniques. Int. J. Image Graph. Signal Process. 2019, 11, 1.

[4] Abdusalomov, A.; Mukhiddinov, M.; Djuraev, O.; Khamdamov, U.; Whangbo, T.K. Automatic Salient Object Extraction Based on Locally Adaptive Thresholding to Generate Tactile Graphics. Appl. Sci. 2020, 10, 3350.

[5] Yoon, H.; Kim, B.H.; Mukhriddin, M.; Cho, J. Salient region extraction based on global contrast enhancement and saliency cut for image information recognition of the visually impaired. KSII Trans. Internet Inf. Syst. (TIIS) 2018, 12, 2287–2312.

[6] Nasir, I.M.; Bibi, A.; Shah, J.H.; Khan, M.A.; Sharif, M.; Iqbal, K.; Nam, Y.; Kadry, S. Deep learning-based classification of fruit diseases: An application for precision agriculture. CMC-Comput. Mater. Contin. 2021, 66, 1949–1962.

[7] Naranjo-Torres, J.; Mora, M.; Hernández-García, R.; Barrientos, R.J.; Fredes, C.; Valenzuela, A. A review of convolutional neural network applied to fruit image processing. Appl. Sci. 2020, 10, 3443.

[8] Macanhã, P.A.; Eler, D.M.; Garcia, R.E.; Junior, W.E.M. Handwritten feature descriptor methods applied to fruit classification. In Information Technology-New Generations; Springer: Berlin/Heidelberg, Germany, 2018; pp. 699–705.

[9] Siddiqi, R. Fruit-classification model resilience under adversarial attack. SN Appl. Sci. 2022, 4, 1–22.

[10] Ukwuoma, C.C.; Zhiguang, Q.; Bin Heyat, M.B.; Ali, L.; Almaspoor, Z.; Monday, H.N. Recent Advancements in Fruit Detection and Classification Using Deep Learning Techniques. Math. Probl. Eng. 2022, 2022, 9210947.

[11] Hossain, M.S.; Al-Hammadi, M.; Muhammad, G. Automatic fruit classification using deep learning for industrial applications. IEEE Trans. Ind. Inform. 2018, 15, 1027–1034.

[12] Rojas-Aranda, J.L.; Nunez-Varela, J.I.; Cuevas-Tello, J.C.; Rangel-Ramirez, G. Fruit classification for retail stores using deep learning. In Mexican Conference on Pattern Recognition; Springer: Berlin/Heidelberg, Germany, 2020; pp. 3–13.