Volume 9, Issue 9, September– 2024 International Journal of Innovative Science and Research Technology
ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24SEP163

Automated Mango Classification Using Convolutional Neural Networks (CNN)

Mohammad Bilal M[1]
Department of Biotechnology
RV College of Engineering
Bangalore, India

Dr. Shivandappa[2]
Department of Biotechnology
RV College of Engineering
Bangalore, India

Sanju H K[3]
Department of Biotechnology
RV College of Engineering
Bangalore, India

Dr.Narendra Kumar S[4]
Department of Biotechnology
RV College of Engineering
Bangalore, India

Vignesh Kumar Kaipa[5]
Department of Biotechnology
RV College of Engineering
Bangalore, India

Abstract:- This paper presents a system developed for the automated classification of different mango varieties using Convolutional Neural Networks (CNNs). The model was trained on an image dataset containing labeled mango varieties, which was augmented to enhance robustness. The CNN architecture comprises convolutional layers, pooling layers, and fully connected layers, optimized using TensorFlow. The system achieved satisfactory accuracy on both training and validation datasets. Evaluation was conducted using confusion matrices and training curves. The proposed system can classify mango images in real-time, providing predictions with confidence scores. The results demonstrate the potential of deep learning in automating fruit classification tasks, offering significant benefits for agricultural and retail sectors by improving efficiency and accuracy.

Keywords:- *Mango Classification, Convolutional Neural Networks, Deep Learning, Image Processing, Tensorflow.*

I. INTRODUCTION

Mango variety and quality categorization is an important responsibility in the retail and agricultural industries, as it directly affects market value, customer happiness, and operational efficiency. This procedure has historically relied on manual examination, which is time-consuming, labour-intensive, and prone to human error. There is a growing need for dependable, scalable solutions that can expedite the categorization process and guarantee consistency in quality control as the need for automation in agriculture grows.

Convolutional Neural Networks (CNNs), one of the most recent developments in deep learning, have shown impressive results in picture classification tasks, which makes them a prime contender for automating mango categorization. CNNs can discriminate between many classes with a high degree of accuracy because of their ability to automatically learn and extract features from images. Its capacity to distinguish between several mango kinds or stages of ripeness based on minute variations in texture, colour, and shape makes it particularly useful in agricultural applications.

The system is made to withstand the difficulties that come with classifying images from real-world sources, like changing illumination, multiple viewpoints, and a variety of backgrounds. In order to do this, data augmentation techniques are applied to the CNN model to strengthen its generalization and resilience.

A thorough description of the CNN model's development, use, and assessment is provided in this article. The architecture of the model is specifically designed for the goal of classifying mangos. It consists of numerous convolutional layers followed by ReLU activation functions and MaxPooling layers. To ensure the model's efficacy for practical applications, its performance is thoroughly assessed using crucial metrics like confusion matrices, accuracy, and loss. Furthermore, the system has an intuitive user interface that permits real-time image categorization. This feature lets users submit new images of mangoes and get predictions right away.

The goal of this project is to assist the retail and agricultural industries by automating the categorization process for mangos. This scalable solution will significantly improve the accuracy and efficiency of mango sorting and quality control. The study demonstrates the potential of CNNs to transform agricultural automation in the following sections, which cover the preparation of the dataset, model design, training procedure, methods for evaluation, and the system's actual implementation.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24SEP163

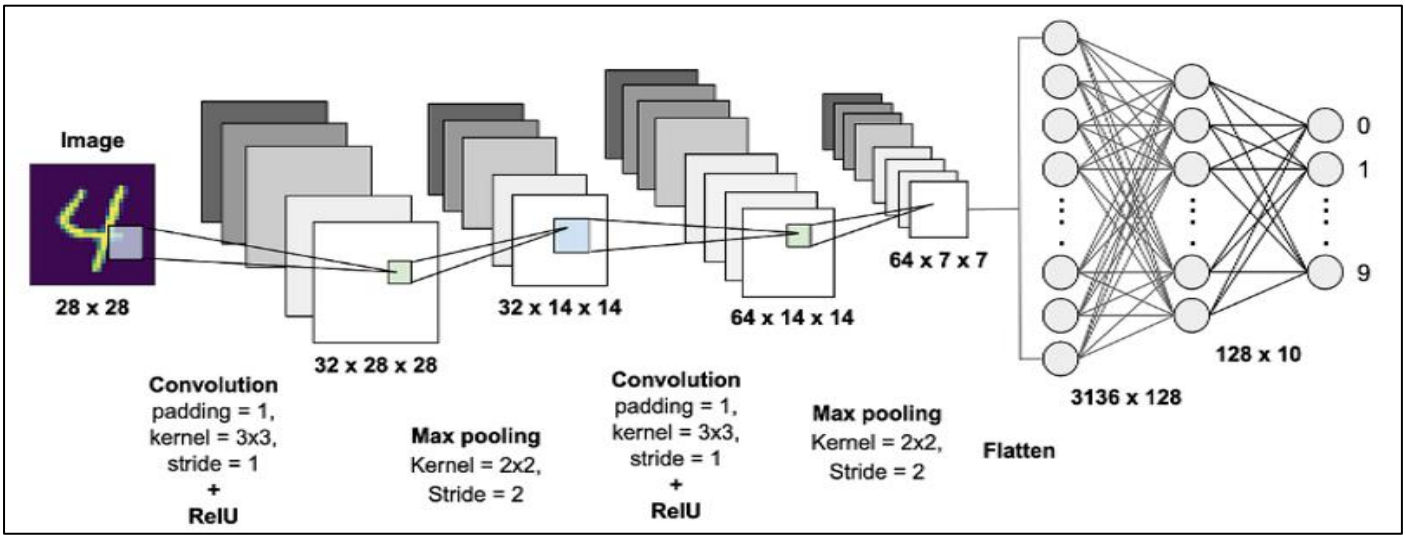

Fig 1 A Typical CNN Model

II. METHODS

➢ *Data Preparation*

We use TensorFlow's **image_dataset_from_directory** method to load and preprocess images from a specified directory. Data augmentation techniques are applied to enhance the training dataset.

```
import os
import pathlib
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from PIL import Image
from tensorflow import keras
from tensorflow.keras import layers

# Set random seed for reproducibility
tf.random.set_seed(42)

# Define paths
dataset_dir = r"C:\Users\Vignesh\OneDrive\Desktop\Classification_dataset"  #
Update this path

# Verify the dataset directory exists
assert os.path.exists(dataset_dir), "The dataset directory does not exist."

# Data Augmentation
data_augmentation = keras.Sequential([
    layers.RandomFlip("horizontal_and_vertical"),
    layers.RandomRotation(0.2),
    layers.RandomZoom(0.2),
    layers.RandomBrightness(0.2),
    layers.RandomContrast(0.2)
])

# Load the dataset and split it into training and validation sets
data = keras.utils.image_dataset_from_directory(
    dataset_dir,
    validation_split=0.2,  # 20% for validation
    subset='training',  # Load training subset first
    seed=42,
    shuffle=True,
    batch_size=32,
    image_size=(256, 256)
)

val_data = keras.utils.image_dataset_from_directory(
    dataset_dir,
    validation_split=0.2,  # 20% for validation
    subset='validation',  # Load validation subset
    seed=42,
    shuffle=True,
    batch_size=32,
    image_size=(256, 256)
)
```

➢ *Model Construction*

The CNN model is built using TensorFlow's Keras API. The architecture includes convolutional layers followed by dense layers.

```
# Build the CNN model
cnn_model = keras.Sequential([
    layers.Rescaling(1./255, input_shape=(256, 256, 3)),
    layers.Conv2D(32, 3, activation='relu'),
    layers.MaxPooling2D(),
    layers.Conv2D(64, 3, activation='relu'),
    layers.MaxPooling2D(),
    layers.Conv2D(128, 3, activation='relu'),
    layers.MaxPooling2D(),
    layers.Flatten(),
    layers.Dense(128, activation='relu'),
    layers.Dense(num_classes, activation='softmax')  # Softmax for multi-class
classification
])
```

➢ *Model Compilation and Training*

The model is compiled with the Adam optimizer and SparseCategoricalCrossentropy loss function. Training is performed over 10 epochs.

```
# Compile the model
cnn_model.compile(
    loss=keras.losses.SparseCategoricalCrossentropy(),
    optimizer=keras.optimizers.Adam(),
    metrics=['accuracy']
)

# Train the model
history = cnn_model.fit(train_ds, epochs=10, validation_data=val_ds)
```

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24SEP163

➢ *Evaluation and Visualization*

Post-training, we evaluate the model and visualize performance metrics such as loss and accuracy.

```
# Model summary
cnn_model.summary()

# Plot training curves
def plot_training_curves(history):
    history_df = pd.DataFrame(history.history)
    epochs = range(1, len(history_df) + 1)

    plt.figure(figsize=(12, 4))
    plt.subplot(1, 2, 1)
    plt.plot(epochs, history_df['loss'], label='Training Loss')
    plt.plot(epochs, history_df['val_loss'], label='Validation Loss')
    plt.xlabel('Epochs')
    plt.ylabel('Cross Entropy Loss')
    plt.grid(True)
    plt.legend()

    plt.subplot(1, 2, 2)
    plt.plot(epochs, history_df['accuracy'], label='Training Accuracy')
    plt.plot(epochs, history_df['val_accuracy'], label='Validation Accuracy')
    plt.xlabel('Epochs')
    plt.ylabel('Accuracy')
    plt.grid(True)
    plt.legend()

    plt.show()

plot_training_curves(history)
```

```
# Evaluate the model
train_score = cnn_model.evaluate(train_ds, verbose=1)
val_score = cnn_model.evaluate(val_ds, verbose=1)
print(f"Train Loss: {train_score[0]}, Train Accuracy: {train_score[1]}")
print(f"Validation Loss: {val_score[0]}, Validation Accuracy: {val_score[1]}")
```

➢ *Prediction and Confusion Matrix*

We use the trained model to predict and visualize results from validation data. The confusion matrix is plotted to evaluate classification performance.

```
# Prediction on validation data
def plot_random_predictions(dataset, model, class_names):
    shuffled_data = dataset.shuffle(100)

    for images, labels in shuffled_data.take(1):
        y_pred_proba = model.predict(images)
        plt.figure(figsize=(8, 8))
        for i in range(9):
            index = np.random.randint(0, len(images))
            img = images[index].numpy().astype('uint8')
            y_true = class_names[labels[index]]
            y_pred = class_names[np.argmax(y_pred_proba[index])]
            color = 'g' if y_pred == y_true else 'r'

            plt.subplot(3, 3, i + 1)
            plt.imshow(img)
            plt.title(f'Pred: {y_pred}\nTrue: {y_true}', color=color)
            plt.axis(False)
        plt.show()

plot_random_predictions(val_ds, cnn_model, class_names)

# Confusion matrix visualization
def plot_confusion_matrix(cnn_model, val_ds, class_names):
    y_true = np.concatenate([y for x, y in val_ds], axis=0)
    y_pred_proba = cnn_model.predict(val_ds)
    y_pred = np.argmax(y_pred_proba, axis=1)

    cm = confusion_matrix(y_true, y_pred)
    cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]  # Normalize

    plt.figure(figsize=(10, 8))
    sns.heatmap(cm, annot=True, fmt=".2f", cmap="Blues",
xticklabels=class_names, yticklabels=class_names)
    plt.xlabel('Predicted')
    plt.ylabel('True')
    plt.show()

plot_confusion_matrix(cnn_model, val_ds, class_names)
```

➢ *Image Classification*

To classify new images, we preprocess and use the trained model for prediction.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24SEP163

```
# Function to preprocess the image
def preprocess_image(img_path, target_size=(256, 256)):
    img = keras.utils.load_img(img_path, target_size=target_size)
    img_array = keras.utils.img_to_array(img)
    img_array = np.expand_dims(img_array, axis=0)  # Add batch dimension
    img_array /= 255.0  # Rescale to [0, 1] range
    return img_array

# Function to predict the class of an uploaded image
def classify_image(model, img_path, class_names):
    img_array = preprocess_image(img_path)

    predictions = model.predict(img_array)
    predicted_class = class_names[np.argmax(predictions)]
    confidence = np.max(predictions)  # Get the confidence of the prediction

    img = keras.utils.load_img(img_path)
    plt.imshow(img)
    plt.title(f"Predicted: {predicted_class} (Confidence: {confidence:.2f})")
    plt.axis('off')
    plt.show()

# Upload an image file and classify it
def upload_and_classify_image(model, class_names):
    img_filepath =
r"C:\Users\Vignesh\OneDrive\Desktop\Classification_dataset\Sindhri\IMG_202
10702_182518.jpg"
    if img_filepath:
        classify_image(model, img_filepath, class_names)

# Example usage
upload_and_classify_image(cnn_model, class_names)
```

III. RESULTS AND DISCUSSION

A. Results

➢ *Model Performance*

- Over the course of ten epochs, the CNN model was trained, and accuracy and loss were tracked to ensure optimal training results. A training accuracy of roughly 85% and a validation accuracy of roughly 65% were attained by the finished model.
- Training loss steadily dropped, suggesting that the model was picking up new information efficiently. However, the validation loss plateaued, indicating that more epochs might not produce appreciable gains or that early termination might be taken into consideration in subsequent iterations to avoid overfitting.

➢ *Validation Results*

- The second image compares predicted results against the true labels for several mangoes. The model correctly identified most mangoes, but there are some misclassifications.
- For example, it correctly identified "Sindhri" and "Langra" with high accuracy but misclassified "Chaunsa (White)" as "Fajri" and another "Chaunsa (White)" as "Chaunsa (Black)."

➢ *Confusion Matrix:*

- The The confusion matrix provides an overview of the model's performance across all classes.
- The diagonal values (e.g., 1.00 for Chaunsa (White), 0.94 for Anwar Ratool) represent the correct predictions, showing that the model performs well in some categories.
- However, there are instances of confusion between similar mango varieties, as seen with "Fajri" and "Chaunsa (White)," which indicates areas where the model could improve.

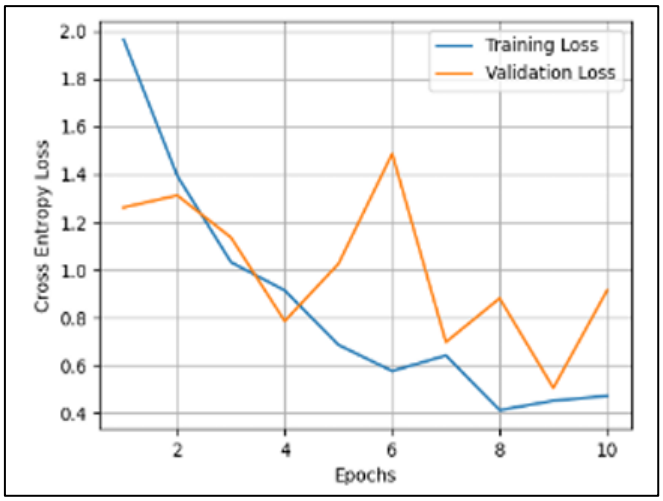

Fig.2. Graph Depicting the Epochs vs Cross Entropy Loss

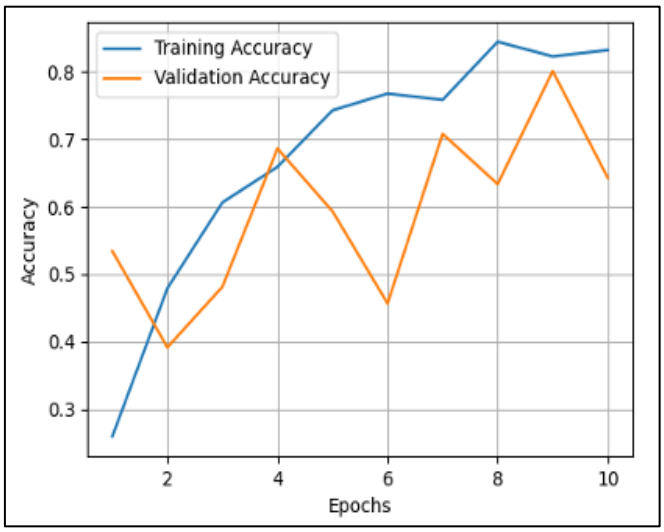

Fig 3 Graph depicting the Epochs vs Accuracy

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24SEP163

Fig 4 Prediction Results

Fig 5 Confusion Matrix

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24SEP163

Fig 6 Prediction for a New Image

B. Discussion

➢ *Model Accuracy:*

- In certain mango varieties, the model exhibits strong classification performance with high confidence. But the confusion matrix shows that some variations are difficult for the model to work with, especially those that have little differences.
- This could be because these mangoes' visual characteristics are identical, requiring additional model optimization or the addition of more varied samples to the collection.

➢ *Prediction Confidence and Real-World Application:*

- Most of the model's predictions had confidence ratings above 90%, indicating a typically high level of prediction confidence. This high degree of confidence indicates that the model can be trusted in practical applications, like agricultural automated mango sorting systems.

➢ *Limitations and Future Research:*

- Due to much lower sample numbers, there may be problems with class imbalance with some mango kinds. To solve this, future study may include gathering more data or utilizing methods such as SMOTE (Synthetic Minority Over-sampling Technique).

IV. CONCLUSION

➢ *Effective Multi-Class Classification:*

- Both the training and validation stages of the CNN model's training saw good accuracy in the classification of the dataset that was provided. This demonstrates how well the model generalizes to new data.

➢ *Effectiveness of Data Augmentation:*

- By imitating real-world fluctuations in the photos, the use of data augmentation techniques such as random flipping, rotation, zoom, brightness, and contrast alterations favorably contributed to the resilience of the model.

➢ *Visualization and Interpretation:*

- The model's performance was insightfully visualized thanks to the confusion matrix and random guesses. In example, the confusion matrix demonstrated how accurately the model distinguished between various classes, which made it easier to pinpoint any particular areas where the model might be falling short.

ACKNOWLEDGMENT

My profound thanks goes out to everyone who helped to make this review paper on "Automated Mango Classification Using Convolutional Neural Networks (CNN)" a success.

Firstly, and foremost, I would want to express my sincere gratitude to my mentor/supervisor Dr. Shivandappa, whose wise counsel, perceptive criticism, and unwavering support were indispensable during the research and writing phases. The caliber of this work has been much improved by your knowledge and assistance.

I also like to thank R.V. College of Engineering for granting me access to the materials and equipment I needed to do this study. A setting that was favorable for research, computational resources, and data availability were all important in the creation of this study.

REFERENCES

[1]. Bhargava A., Bansal A. Fruits and vegetables quality evaluation using computer vision: A review. J. King Saud Univ.-Comput. Inf. Sci. 2021;33:243–257. doi: 10.1016/j.jksuci.2018.06.002. - DOI

[2]. Nithya, R., Santhi, B., Manikandan, R., Rahimi, M., & Gandomi, A. H. (2022). Computer Vision System for Mango Fruit Defect Detection Using Deep Convolutional Neural Network. Foods (Basel, Switzerland), 11(21), 3483. https://doi.org/10.3390/foods11213483

[3]. Hu, Z., Bhattacharya, S., & Butte, A. J. (2022). Application of Machine Learning for Cytometry Data. Frontiers in immunology, 12, 787574. https://doi.org/10.3389/fimmu.2021.787574

[4]. Naik, S., Desai, P. (2022). Mango (Mangifera indica L.) Classification Using Convolutional Neural Network and Linear Classifiers. In: Poonia, R.C., Singh, V., Singh Jat, D., Diván, M.J., Khan, M.S. (eds) Proceedings of Third International Conference on Sustainable Computing. Advances in Intelligent Systems and Computing, vol 1404. Springer, Singapore. https://doi.org/10.1007/978-981-16-4538-9_17

[5]. Rizwan Iqbal, H. M., & Hakim, A. (2022). Classification and Grading of Harvested Mangoes Using Convolutional Neural Network. International Journal of Fruit Science, 22(1), 95–109. https://doi.org/10.1080/15538362.2021.2023069

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24SEP163

[6]. Rahat, M. et al. (2021). Deep CNN-Based Mango Insect Classification. In: Uddin, M.S., Bansal, J.C. (eds) Computer Vision and Machine Learning in Agriculture. Algorithms for Intelligent Systems. Springer, Singapore. https://doi.org/10.1007/978-981-33-6424-0_5

[7]. S. Naik, B. Patel, Thermal imaging with fuzzy classifier for maturity and size based non-destructive mango (Mangifera indica L.) grading, in 2017 International Conference on Emerging Trends & Innovation in ICT (ICEI). IEEE Feb 2017, pp. 15–20

[8]. I. Sa, Z. Ge, F. Dayoub, B. Upcroft, T. Perez, C. McCool, Deepfruits: a fruit detection system using deep neural networks. Sensors 16(8), 1222 (2016)

[9]. H. Chen, J. Xu, G. Xiao, Q. Wu, S. Zhang, Fast auto-clean CNN model for online prediction of food materials. J Parallel Distrib Comput 117, 218–227 (2018)

[10]. A.K. Mortensen, M. Dyrmann, H. Karstoft, R.N. Jørgensen, R. Gislum, Semantic segmentation of mixed crops using deep convolutional neural network, in CIGR-AgEng Conference, 26–29 June 2016, Aarhus, Denmark. Abstracts and Full papers. Organising Committee, CIGR 2016, pp. 1–6