Team members: Zicong He, Ming Zhu, Ruilan Sun, Hongyu Guo

1. Introduction

In the fields of machine learning and computer vision, the classification of objects has always been a topic of extensive research and practical application. Among various objects, fruits hold a special place due to their diversity in shape, color, and texture. Accurate fruit classification is crucial in multiple industries, including agriculture, retail, and health. Automated classification systems can assist in quality control, inventory management, and nutritional analysis, making them indispensable in the modern agricultural economy and health-conscious consumer markets. However, the challenge lies in accurately identifying and classifying a wide variety of fruits, as their appearance can vary due to factors like lighting, orientation, and natural diversity. To address this challenge, we explore the application of Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), implementing and evaluating these two distinct neural network architectures for fruit classification.

The Fruit-360 dataset, known for its diverse range of fruit images, provides an excellent foundation for examining the efficacy of these models in distinguishing fruit types based on visual features. Notably, the dataset includes separate testing and training sections, eliminating the need for additional steps to split the dataset into training and testing subsets.
(The total number of images: 90483.
Training set size: 67692 images (one fruit or vegetable per image).
Test set size: 22688 images (one fruit or vegetable per image).
The number of classes: 131 (fruits and vegetables).
Image size: 100x100 pixels.)

Whether for CNN or RNN, the first step involves data preprocessing, which is divided into three stages. The first is image size adjustment: all images are resized to 100x100 to ensure consistency in input size. The second step is dimension transpose, converting images from Height x Width x Channels (HWC) format to Channels x Height x Width (CHW) format, optimizing for best processing and feature extraction from image data. The final critical step is normalization, scaling pixel values to a standard range (usually 0-1 or -1 to 1). This step is vital as it brings consistency to input value ranges, aiding in stabilizing the learning process and accelerating model convergence during training. It also helps reduce issues related to lighting differences in images.

2.1 CNN—Implementation & Architecture:

Our CNN model, defined using PaddlePaddle, features a series of convolutional, pooling, and fully connected layers. It starts with a Conv2D layer with 16 filters, a kernel size of 5, and 'SAME' padding, followed by a MaxPool2D layer with a kernel size of 2 and a stride of 2. This pattern is repeated with an increasing number of filters in subsequent convolutional layers (32, 64, and finally 128). Each convolutional layer is succeeded by a ReLU activation function to introduce non-linearity and a max pooling layer to reduce spatial dimensions and enhance feature extraction.

The output from the final pooling layer is flattened into a one-dimensional tensor and fed into a fully connected linear layer with 256 neurons. A dropout layer with a dropout rate of 0.8 follows the first fully connected layer to prevent overfitting. The network concludes with an output layer, a fully connected linear layer with 131 neurons, corresponding to the number of fruit classes in the Fruit-360 dataset.

In the forward method, our CNN processes the input tensor x through its layers. The input first passes through the convolutional and pooling layers, where it is transformed and downscaled, extracting and amplifying relevant features. The flattened output then passes through the linear, ReLU activation, dropout, and final output layers.

In summary, our model is a robust CNN designed for image classification tasks. It combines convolutional layers for feature extraction, pooling layers for dimensionality reduction, fully connected layers for classification, and a dropout layer for regularization. The use of ReLU activation functions adds the necessary non-linearity to the model, enhancing its classification capabilities.

2.2 RNN—Implementation & Architecture:
We define the RNN neural network using PaddlePaddle, a deep learning framework. Let's analyze its architecture and implementation of this deep learning system.

Architecture:
Our model initializes an LSTM (Long Short-Term Memory) layer with 128 hidden units, and consists of two stacked LSTM layers. Each input feature vector has 300 elements. Followed by a fully connected linear layer that maps from 128 features (output from LSTM) to 256 features. We also added a dropout layer: a regularization layer that randomly zeros some of the elements of the input tensor with probability 0.5 during training, to prevent overfitting. Finally, the output layer is another fully connected linear layer that maps from 256 features to 131 output classes.

Implementation:
We use the forward method to define the forward pass of the network. It processes the input x through the network layers. For each element in the batch, it reshapes and transposes the input tensor to match the expected input dimensions for the LSTM layer. After processing the input through the LSTM layer, we extract the last output of the sequence for each batch element. The outputs from all batch elements are concatenated. The concatenated tensor is then passed through the linear, ReLU activation, dropout, and the final output layer.
-In summary, the model is a typical LSTM-based neural network for processing sequential data. It includes an LSTM layer for capturing temporal dependencies, linear layers for transformation, a dropout layer for regularization, and uses ReLU activation.

3.1 CNN—Training Strategy and Reasoning, Challenge and Solution:
In the initial stages of our experiment, we deployed a Convolutional Neural Network (CNN) for fruit classification using the Fruit-360 dataset. Initially, the model showed signs of overfitting: while it achieved a high accuracy of 99% on the training dataset, the accuracy on the test dataset was significantly lower at only 92%. This discrepancy suggested that the model was too closely

tailored to the training data and struggled to generalize to new, unseen data.

To tackle this issue, we made critical enhancements to the architecture of the CNN. The introduction of additional Pooling Layers and Dropout Layers was pivotal in addressing overfitting. Pooling layers, particularly MaxPooling layers, helped in reducing the spatial dimensions of the feature maps, thus minimizing the risk of overfitting by reducing the number of learnable parameters. Meanwhile, Dropout Layers randomly deactivated a fraction of the neurons during training, which prevented the model from becoming overly reliant on specific features or patterns in the training data.

3.2 RNN—Training Strategy and Reasoning, Challenge and Solution:
RNN:
In the initial stages of our experiment, we employed a standard RNN (Recurrent Neural Network) model for the task of fruit classification. However, the performance of this model was not satisfactory. Despite increasing the number of training epochs and fine-tuning the learning rate, we encountered significant challenges. The most prominent issue was the slow convergence rate of the model during training. Furthermore, the validation accuracy achieved by the RNN model was disappointingly low, indicating a potential inability of the model to capture the complex patterns in the dataset effectively.

To address these limitations, we shifted our approach to a more advanced variant of recurrent neural networks: the LSTM (Long Short-Term Memory) network. The LSTM architecture is renowned for its ability to capture long-term dependencies and mitigate issues like vanishing gradients, which are common in traditional RNNs. This change in the model architecture led to substantial improvements in our training outcomes.

4.1 CNN—Conclusion:
The introduction of Pooling and Dropout layers proved to be highly effective. Following these adjustments, the model not only maintained high accuracy on the training dataset but also showed significant improvement in test dataset performance. The test accuracy increased notably, reaching 97.05%. This enhancement in test accuracy indicated a more robust model that was better at generalizing and adapting to new data.

In summary, an iterative process of improving CNN architecture, as well as a keen focus on preventing overfitting, led to the development of a more balanced and versatile model. The model expertly handles the challenges posed by the Fruit-360 dataset and shows superior performance in the fruit classification task.

4.2 RNN—Conclusion:
Upon implementing the LSTM-based model, we observed a remarkable acceleration in the convergence rate. The model successfully converged after 30 epochs of training, which was a significant reduction compared to the earlier RNN model. More importantly, this rapid convergence did not come at the cost of accuracy. In fact, the LSTM model demonstrated a high validation accuracy of 92.23%, indicating its superior capability in generalizing and learning from

the fruit classification dataset.

This transition from a standard RNN to an LSTM model underscores the importance of choosing the right architecture for specific tasks in machine learning. The LSTM's ability to handle sequential data and remember information over longer periods made it particularly suitable for the complexities of fruit classification, where the temporal sequence of the data plays a crucial role.

4.3 General Conclusion

In this project, whether for CNN or RNN, we tried various methods to improve accuracy and generalizability, and to reduce the likelihood of overfitting, such as adding Pooling and Dropout layers in the CNN, and transitioning from a traditional RNN to an LSTM-style RNN.

In conclusion, our experimental results demonstrate a marked superiority of Convolutional Neural Networks (CNNs) over Recurrent Neural Networks (specifically LSTM models) in the context of our fruit classification task. This conclusion is drawn from several key observations. Firstly, both CNN and LSTM architectures achieved similar levels of accuracy in classifying fruits. However, the rate at which these models reached their peak performance varied significantly. The CNN model exhibited a remarkably faster convergence, reaching optimal performance within just 5 epochs of training. In stark contrast, the LSTM model required a considerably longer training period, converging only after 30 epochs. This significant difference in convergence speed is a critical factor in the context of practical applications where training time and computational efficiency are of essence.

The superior performance of CNNs in this specific task can be attributed to several inherent characteristics of these models. CNNs are particularly well-suited for image recognition and classification tasks, owing to their ability to efficiently process and extract hierarchical features from spatial data like images. Their architectural design, which includes convolutional layers and pooling layers, allows them to capture spatial hierarchies in data by applying filters that recognize patterns and features at various levels of abstraction.

On the other hand, while LSTMs are adept at handling sequential data and capturing long-term dependencies, their structure is not naturally aligned with the demands of image classification tasks. Images, particularly those in our fruit classification dataset, are characterized by their spatial relationships and localized features like textures and shapes. LSTMs, with their sequential processing approach, are less efficient at capturing these spatial hierarchies intrinsic to image data.

In light of these observations, it is evident that the choice of model architecture should be closely aligned with the nature of the task at hand. For tasks involving spatial data, such as image classification, CNNs are evidently more suitable due to their ability to efficiently process and learn from the inherent structure of the data. This experiment serves as a clear testament to the importance of aligning model architecture with the specific characteristics of the dataset and task objectives in the field of machine learning.

However, in the end, we realized that the generalizability of both the CNN and RNN models was not exceptionally high, with only 30-40% accuracy in out-of-domain scenarios. Upon careful investigation, we found that the data set was too small, leading to many fruit images being blurred or too uniform. Nevertheless, the accuracy achieved on the test set represents a significant improvement, and we anticipate further enhancement in the models' generalizability as the dataset becomes more comprehensive in the future.