Fruit Classification using Convoluted Neural Network

Introduction

The aim of this assignment was to develop a Convoluted Neural Network model which can be used to classify fruit images into classes: Grape, Apple, Mango, Banana, or Strawberry. TensorFlow and Keras were used to create a CNN comprised of 3 convolution layers, max pooling layers, flattens, and dense layers. The model was trained on the Fruits data set from Kaggle and was evaluated based on accuracy, precision, recall, history plots, and figure that presents the accuracy vs Epoch.

Methodology

Dataset Description

The Fruits data set contains images of 5 fruits: Grape, Apple, Mango, Banana, or Strawberry. Each class contains 2000 images, resulting in a total of 10,000 images in the dataset. The images in the dataset are of various shapes, sizes, and colors, and have been captured under different lighting conditions. The data is split into three sets: training, validation, and testing. The dataset is split based on a ratio of 97% for training, 2% for validation, and 1% for testing. Each class in the dataset is split into three sets based on the ratio, so 97% (1940 images) are used for training, 2% (40 images) are used for validation, and 1% (20 images) are used for testing in each of the five classes.

Data Pre-processing

To improve accuracy and reliability of the model preprocessing steps were completed before bringing the dataset into the model. As we were working with images, all instances needed to be standardized. ImageDataGenerator from Keras was used to resize each image to 98 by 98 pixels. The images were also rescaled to values between 0 and 1.

```
img_width, img_height = 98, 98
input_shape = (img_width, img_height, 3)
```

```
datagen = ImageDataGenerator(
    rescale=1./255)
```

As the dataset was already sorted into train, validation, and test models, those were left alone. However, batches of data were pre-portioned out to provide to the model which also includes the resizing.

```
train_gen = datagen.flow_from_directory(
    train_path,
    target_size=(img_width, img_height),
    batch_size=64,
    class_mode='categorical')
```

Model

The model used a convoluted neural network comprised of three convolution layers made using TensorFlow and Keras. These convolution layers have increasing filter sizes (32, 64, 128) for feature extraction, and all three have a kernel size of 3x3 pixels. The increase in filter size aims to allow the model to identify increasingly complex patterns and features. Each convolution layer uses a Rectified Linear Unit (ReLu) activation function which was chosen to introduce non-linearity and express any complexities available within the data. Every convolution layer is followed by a Max Pooling layer with a 2x2 window size to reduce the spatial dimensions by half

while maintaining relevant information in the feature maps. This reduces the computation needs of the model, making it more efficient.

After the three convolution, max pooling layers, a Flatten Layer in included to reduce the output into a one dimensional vector feature map, rather than an image. After flattening, two fully connected dense layers were used for classification. The first dense layer had 512 units with ReLU activation for use as a hidden layer. The second layer had 5 units corresponding to the five fruit classes with softmax activation used for the final classification.

```
model = Sequential([
    Conv2D(filters=32, kernel_size=(3,3), activation='relu', input_shape=input_shape),
    MaxPooling2D(2, 2),

    Conv2D(64, (3, 3), activation='relu'),
    MaxPooling2D(2, 2),

    Conv2D(128, (3, 3), activation='relu'),
    MaxPooling2D(2, 2),

    Flatten(),

    Dense(512, activation='relu'),
    Dense(5, activation='softmax') #number of classes
```

The model was compiled with categorical crossentropy loss and the Adam optimizer, using accuracy as the metric.

```
model.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=['accuracy'])
```

Hyperparameters

Hyperparameters are crucial for tuning and improving model performance. The parameters of interest to this model are the number of neurons in the hidden layer, number of epochs, and the batch size. As well as evaluation metrics and validation split. The values for the hyperparameters are below.

Hyperparameter	**Value**
Filters	32, 64, 128
Number of epochs	200
Batch size	32, 64
Evaluation Metric	Accuracy

The number of filters used in the convolutional layers is set to 32, 64, and 128, respectively, in the three convolutional layers. These values will allow for exploration of complex features. The model is trained for a total of 200 epochs, allowing it to learn from the training data over multiple iterations. Two batch sizes, 32 and 64, are utilized during the training process, and were chosen to allow for some complexity without being too computationally intense. The primary evaluation metric used to assess the model's performance is accuracy as we are concerned with correctly identifying the fruit.

```
history = model.fit(train_gen,
                    epochs=200,
                    validation_data=val_gen)
```

Evaluation Metrics

To evaluate the efficacy of the model several metrics were used for each fruit class as well as the overall score for the full dataset. Accuracy was used to identify the percentage of correctly classified samples. Precision to identify the percentage of true positive predictions among all

positive predictions. Recall to identify the proportion of true positive predictions among all actual positive samples. A figure that presents the accuracy vs Epoch was used to visualize the performance of the model in terms of accuracy given the Epoch. Log loss was also used to quantify model performance.

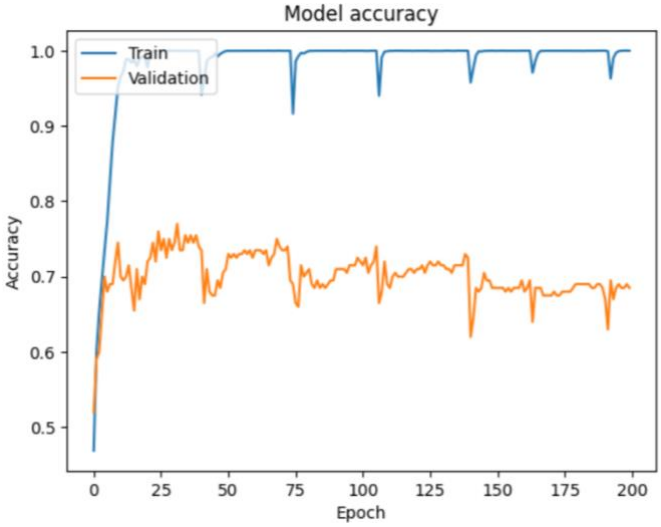
Results

The evaluation metrics obtained from the fruit classification model demonstrate its performance on the dataset. The overall accuracy of the model was acceptable at 75% and compared to the Top-1 Accuracy scores of the paper. However, the fine-tuned models from the paper were able to achieve an accuracy of 99%+, far exceeding the performance of this model. Additionally, the log loss score was somewhat high at 3.16, and lower values are desirable for categorical cross-entropy loss.

```
: loss, accuracy = model.evaluate(test_gen)
4/4 [==============================] - 0s 49ms/step - loss: 3.1603 - accuracy: 0.7500
```

The evaluation metrics for the individual classes are not as promising. This raises the question of if I used the classification report correctly. We see un-acceptable values for all precision, recall, and f1-score for all classes. Precision values range from 0.07 to 0.27, indicating the model's weak ability to accurately identify instances of specific fruit classes. Similarly, recall values vary from 0.05 to 0.30, identify the model's in-effectiveness in capturing instances of each fruit class from the dataset. The F1-scores, balancing precision and recall, range from 0.06 to 0.26, again displaying poor model performance's in correctly identifying instances of each class. However, as the overall accuracy score of 75% for the whole model raises questions about the validity of the classification report.

	precision	recall	f1-score	support
Apple	0.23	0.30	0.26	20
Banana	0.17	0.20	0.19	20
Grape	0.27	0.20	0.23	20
Mango	0.18	0.20	0.19	20
Strawberry	0.07	0.05	0.06	20



The figure that presents the accuracy vs Epoch identifies several peaks and valleys with accuracy from Epoch 1 to 200. About every 25 Epochs a local minimum would occur followed by a local maximum.We can see in the figure that accuracy was highest around epoch 30 where the accuracy is around 77%.Overall the accuracy score is between 60% to 77% over the 200 Epochs.

Conclusion

In conclusion, the developed Convoluted Neural Network model exhibits the lower limit of acceptable performance when classifying types of fruit. With an accuracy score of 75%, but with untrustworthy precision, and recall scores, the model could be used a starting point to for classification. The performance of this model is disappointing when compared to the models in “Comparative Performance of Various Deep Learning based Models in Fruit Image Classification” (Raheel Siddiqi 2020). Improvements and tuning would need to be explored to improve the performance before this model would be able to be used for classification.