Model Optimization and Tuning Phase Template

Date	15 March 2024
Team ID	LTVIP2024TMID24981
Project Title	Deep learning techniques for breast cancer prediction
Maximum Marks	10 Marks

Model Optimization and Tuning Phase

Optimizing and tuning a Convolutional Neural Network (CNN) for breast cancer detection involves several critical steps to ensure that the model performs well. Here's a structured approach.

Model Architecture

Hyperparameter Tuning

Hyperparameter Tuning Documentation (8

Marks):

Model Architecture Hyperparameters

- **Number of Filters in Convolutional Layers**:
  - In your code, you use 32 filters in both Convolution2D layers. Consider tuning this parameter to explore values like 16, 32, 64, and 128 to see how the model performs with different complexities.
- **Filter Size**:
  - The filter size (3, 3) is common. You can experiment with other sizes like (5, 5) or (1, 1) to observe how the receptive field affects learning.
- **Activation Functions**:
  - The ReLU activation function is a standard choice. You can experiment with alternatives like tanh, sigmoid, or Leaky ReLU to see if they provide better convergence and performance.
- **Pooling Size**:
  - You use a pooling size of (2, 2). This is typical, but consider experimenting with different pooling sizes such as (3, 3) or using GlobalMaxPooling2D.
- **LSTM Units**:
  - In the LSTM layer, you set the number of units to 32. Try varying this number between 16, 32, 64, and 128 to see how it affects the model's ability to capture sequential dependencies.

<table>
<tr><th>Model</th><th>Tuned Hyperparameters</th></tr>
<tr><td>CNN</td><td><pre>#ccreating CNN object
cnn_model = Sequential()
#adding CNN2d layer with 32 neurons of size 3 X 3 to filter images 32 times
cnn_model.add(Convolution2D(32, (3 , 3), input_shape = (X_train.shape[1], X_train.shape[2], X_train.shape[3]), activation = 'relu'))
#max pool layer to collect filtered relevant features from previous CNN layer
cnn_model.add(MaxPooling2D(pool_size = (2, 2)))
#adding another layer with relu activation function
#ReLU helps the first hidden layer receive errors from the last layers to adjust all weights between layers
cnn_model.add(Convolution2D(32, (3, 3), activation = 'relu'))
cnn_model.add(MaxPooling2D(pool_size = (2, 2)))
cnn_model.add(Flatten())</pre></td></tr>
<tr><td>ADDING
LSTM as RNN</td><td><pre>#adding LSTM as RNN layer
cnn_model.add(RepeatVector(2))
cnn_model.add(LSTM(32, activation = 'relu'))#=================adding RNN LSTM
#defining output layer with extra softmax layer which will divide each class prediction into probabilities and the
#class with highest probability will be best prediction and help in enhancing accuracy
cnn_model.add(Dense(units = 256, activation = 'relu'))
cnn_model.add(Dense(units = y_train.shape[1], activation = 'softmax'))
#compile the model
cnn_model.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metrics = ['accuracy'])</pre></td></tr>
<tr><td></td><td></td></tr>
</table>

Final Model Selection Justification (2 Marks):

Final Model	Reasoning
CNN	Convolutional Neural Networks (CNNs) are particularly effective for tasks involving image data, making them a valuable tool in the field of medical imaging, including breast cancer prediction. Here are several reasons why CNNs are used for code optimization in this area: **Feature Extraction**: CNNs automatically learn and extract relevant features from images (like mammograms or histopathological slides) through convolutional layers. **Spatial Hierarchy**: CNNs capture spatial hierarchies in images, meaning they can recognize patterns at various scales. **Transfer Learning**: Pre-trained CNN models can be fine-tuned for breast cancer prediction. This transfer learning approach enables leveraging existing knowledge from large datasets to improve performance on smaller datasets, which is often the case in medical applications. **High Performance**: CNNs have shown superior performance compared to traditional machine learning algorithms in image classification tasks. Their ability to process large amounts of data efficiently makes them suitable for analysing complex medical images. **End-to-End Learning**: CNNs facilitate end-to-end learning, meaning that they can take raw image data and directly produce predictions. This streamlines the workflow, as there's no need for separate preprocessing steps.

	Robustness to Variability: CNNs are robust to variations in imaging conditions, such as differences in lighting, noise. **Integration with Other Data Types**: CNNs can be combined with other forms of data (like genomic or clinical data) to improve prediction accuracy. This multimodal approach can enhance the understanding of breast cancer and improve treatment strategies.