CT Scan Image Classification for COVID-19 Data -Approach

Classifying COVID-19 CT scan images using the ResNet (Residual Network) architecture involves several steps, including data preprocessing, model construction, training, and evaluation.

Data Overview

This dataset contains 1252 CT scans that are positive for SARS-CoV-2 infection (COVID-19) and 1230 CT scans for patients non-infected by SARS-CoV-2, 2482 CT scans in total. These data have been collected from real patients in hospitals from Sao Paulo, Brazil. The aim of this dataset is to encourage the research and development of artificial intelligent methods which are able to identify if a person is infected by SARS-CoV-2 through the analysis of his/her CT scans.

1. Data Preprocessing:

- Collect a dataset of COVID-19 CT scan images along with their corresponding labels (COVID-19 positive or negative).

- Split the dataset into training, validation, and test sets.

- Preprocess the CT scan images by resizing them to a consistent input size, applying normalization to bring pixel values within a certain range (e.g., [0, 1]), and augmenting the data (if required) with techniques like rotation, flipping, and zooming.

2. Model Construction:

- Import the necessary libraries (e.g., TensorFlow, Keras).

- Load the ResNet architecture (e.g., ResNet-50, ResNet-101) pre-trained on a large image dataset (e.g., ImageNet).

- Remove the last fully connected layer and replace it with a new dense layer with the number of output classes as 2 (COVID-19 positive or negative).

- Optionally, you can freeze some of the earlier layers to prevent their weights from being updated during training.

3. Model Compilation:

- Compile the model using an appropriate loss function, an optimizer, and a suitable evaluation metric (e.g., accuracy, precision, recall).

4. Model Training:

- Train the modified ResNet model on the training dataset.

- Use the validation dataset for monitoring the model's performance during training and to prevent overfitting.
- Implement techniques like learning rate scheduling and early stopping to improve training efficiency and prevent overfitting.

5. Model Evaluation and Fine-Tuning:

- Evaluate the trained model using the test dataset to assess its generalization performance.

- Calculate metrics such as accuracy, precision, recall, F1-score, and ROC-AUC to measure the model's performance.

- Generate a confusion matrix to visualize the true positive, true negative, false positive, and false negative predictions

- Fine-tune hyperparameters (learning rate, batch size, etc.) and architectural choices based on the evaluation results to enhance model performance.

6. Interpretation and Visualization:

- Visualize the model's learned features using techniques like gradient-weighted class activation mapping to highlight areas in the CT scan images that contributed most to the classification decision.