SNS COLLEGE OF TECHNOLOGY

Saravanampatti, Coimbatore – 035.

19CST302 - Neural Networks & Deep Learning

Create and deploy neural networks using Tensor Flow for Image data.

By:
Dr. A. Sumithra
ASP / CSE

TensorFlow

TensorFlow is an open-source library developed by Google primarily for deep learning applications. It also supports traditional machine learning. TensorFlow was originally developed for large numerical computations without keeping deep learning in mind. However, it proved to be very useful for deep learning development as well, and therefore Google open-sourced it. TensorFlow accepts data in the form of multi-dimensional arrays of higher dimensions called tensors. Multi-dimensional arrays are very handy in handling large amounts of data.

TensorFlow Architecture

TensorFlow architecture works in three significant steps:

- Data pre-processing - structure the data and brings it under one limiting value
- Building the model - build the model for the data
- Training and estimating the model - use the data to train the model and test it with unknown data

Create a neural network with TensorFlow

- Step 1: Install TensorFlow
- Step 2: Import Required Libraries
- Step 3: Prepare the Dataset
- Step 4: Create the Neural Network Model
- Step 5: Compile the Model
- Step 6: Train the Model
- Step 7: Evaluate the Model

Data Preparation

Data Collection: Gather a dataset of images relevant to your task, ensuring it is labeled appropriately.

Data Preprocessing: Preprocess the images by resizing them to a consistent size, normalizing pixel values, and potentially applying augmentation techniques to increase the diversity of the dataset and improve model generalization.

Data Splitting: Split the dataset into training, validation, and test sets to evaluate the model's performance.

Model Building

- **Model Construction:** Use TensorFlow's high-level APIs, such as Keras, to construct the neural network architecture. Define the layers, including convolutional layers, pooling layers, and fully connected layers, as well as the activation functions and output layer.
- **Compiling the Model:** Compile the model by specifying the loss function, optimizer, and evaluation metrics. Common choices for image classification tasks include categorical cross-entropy loss and the Adam optimizer.

Training

- **Model Training:** Train the neural network using the training dataset. Use TensorFlow's ‘model.fit()’ function to train the model for a specified number of epochs, adjusting hyperparameters such as batch size and learning rate as needed.
- **Monitoring Performance:** Monitor the model's performance on the validation set during training to detect overfitting and adjust model architecture or regularization techniques accordingly.
- **Model Saving:** Save the trained model weights and architecture to disk for later deployment.

Evaluation

- **Model Evaluation:** Evaluate the trained model's performance on the test dataset to assess its accuracy and generalization ability.
- **Performance Metrics:** Calculate relevant performance metrics such as accuracy, precision, recall, and F1-score to measure the model's effectiveness in classifying images.

Deployment

- **Model Exporting:** Export the trained model in a format suitable for deployment, such as TensorFlow's SavedModel format or TensorFlow Lite for mobile deployment.
- **Integration with Applications:** Integrate the trained model into your application or platform, ensuring compatibility with the target environment.
- **Continuous Monitoring:** Monitor the model's performance in production, collecting feedback data and periodically retraining the model to adapt to changes in the data distribution.

Conclusion

Throughout this process, TensorFlow provides a comprehensive ecosystem of tools and resources to streamline the development and deployment of neural networks for image data. By following these steps and leveraging TensorFlow's capabilities, you can create robust image classification models and deploy them effectively in real-world applications.