ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

DAY – 16 **14 July 2025**

Introduction to TensorFlow

TensorFlow is an open-source machine learning (ML) and artificial intelligence (AI) framework developed by Google Brain. It simplifies the process of building, training, and deploying ML models—especially deep learning models—across a variety of platforms.

TensorFlow supports applications in areas such as natural language processing (NLP), computer vision (CV), time series forecasting, and reinforcement learning.

Key Features

1. Scalability
   TensorFlow scales from desktops to mobile and embedded devices. It supports distributed computing for efficient large-scale model training.
2. Comprehensive Ecosystem
   - TensorFlow Core – Low-level API for building and executing computations
   - Keras – High-level API for rapid model development
   - TensorFlow Lite – Lightweight runtime for mobile/embedded devices
   - TensorFlow.js – Run ML models in the browser with JavaScript
   - TFX (TensorFlow Extended) – Tools for production deployment
   - TensorFlow Hub – Pre-trained model repository
3. Automatic Differentiation
   Built-in autograd functionality simplifies backpropagation for training.
4. Multi-language Support
   Though Python-first, TensorFlow also supports C++, Java, and JavaScript.
5. Model Serving & Optimization
   Tools like TensorFlow Serving and Model Optimization Toolkit enable efficient deployment and faster inference.

By : Aditi Tangri URN : 2302460 CRN : 2315004

Architecture Overview

- Tensors: Multi-dimensional arrays (scalars, vectors, matrices, etc.)—the basic data units.
- Graph: A computation graph where nodes represent operations and edges represent tensors.
- Session: Executes the operations defined in the graph (primarily in TensorFlow 1.x; replaced by eager execution in TensorFlow 2.x).

TensorFlow Workflow

1. Train the Model
   Develop and train on a PC or cloud using suitable datasets.
2. Convert the Model
   Use TFLite Converter to convert to .tflite for edge deployment.
3. Optimize the Model
   Apply techniques like quantization or pruning to reduce size and increase speed.
4. Deploy the Model
   Deploy to devices like Android, iOS, Raspberry Pi, or microcontrollers.
5. Run Inference
   Use TFLite Interpreter for real-time, low-latency predictions on-device without cloud dependence.