# **MNIST Neural Network Comparison: TensorFlow vs PyTorch**

This project compares the training and evaluation of neural network models on the MNIST dataset using **TensorFlow** and **PyTorch**. Multiple architectures and configurations are tested to understand the differences in performance, training time, and implementation style.

## **🧠 Dataset**

The MNIST dataset contains 70,000 grayscale images of handwritten digits (0–9), with each image being 28×28 pixels. It is widely used for benchmarking image classification models.

## **📁 Project Structure**

graphql

CopyEdit

mnist\_comparison/  
├── tensorflow\_basic.py  
├── tensorflow\_custom\_rmsprop.py  
├── tensorflow\_custom\_sgd.py  
├── pytorch\_model.py  
├── model.tflite # TensorFlow Lite converted model  
├── model.onnx # ONNX exported PyTorch model  
├── README.md

## **✅ Features**

* **Basic TensorFlow model**: Simple 1-hidden-layer network using ReLU and Adam optimizer.
* **Custom TensorFlow training loop (RMSprop)**: Deep network with Tanh + ReLU layers and RMSprop optimizer.
* **Custom TensorFlow training loop (SGD)**: Deeper model with LeakyReLU and SGD optimizer.
* **PyTorch model**: Multi-layer feedforward network using LeakyReLU and SGD.
* **Model export**:
  + TensorFlow → TFLite
  + PyTorch → ONNX

## **⚙️ Requirements**

Install the required packages with:

bash

CopyEdit

pip install tensorflow torch torchvision onnx

## **🚀 How to Run**

### **1. TensorFlow (Basic)**

bash

CopyEdit

python tensorflow\_basic.py

### **2. TensorFlow (Custom training with RMSprop)**

bash

CopyEdit

python tensorflow\_custom\_rmsprop.py

### **3. TensorFlow (Custom training with SGD)**

bash

CopyEdit

python tensorflow\_custom\_sgd.py

### **4. PyTorch**

bash

CopyEdit

python pytorch\_model.py

## **📈 Outputs**

Each script will display:

* Training time
* Epoch-wise accuracy
* Final test accuracy
* (Optionally) saved model in .tflite or .onnx format

## **🔁 Model Export**

* **TensorFlow to TFLite**: The trained model is saved as model.tflite for deployment on edge devices.
* **PyTorch to ONNX**: The model is exported to model.onnx, allowing inference in other frameworks or platforms.

## **📊 Comparison Criteria**

* **Training Time**
* **Test Accuracy**
* **Model Architecture**
* **Optimizer Performance**
* **Ease of Implementation**

## **📌 Notes**

* Each version uses similar preprocessing steps for fairness.
* Batch sizes, epochs, and optimizers are varied to demonstrate flexibility and performance trade-offs.
* The PyTorch and TensorFlow models are **not identical**, but similar in complexity for a balanced comparison.

## **📄 License**

This project is released under the MIT License.

Let me know if you'd like the README tailored for Jupyter notebooks, or a version that includes plots or visualizations.

Ask ChatGPT

Tools