# **Simulation Documentation**

## **1. Setup**

* **Data Loading**: The MNIST dataset was successfully loaded using tf.keras.datasets.mnist. This is a standard and efficient method for training a model on the MNIST dataset, and it was executed without any issues.
* **Model Creation**: A simple Convolutional Neural Network (CNN) model was created with a few layers for MNIST classification. The model was compiled successfully. However, an issue arose during compilation due to incompatible data types, such as float32 for the input tensors and int64 for the labels, causing errors. This was resolved by ensuring both the input and labels were of the same data type.
* **TFLite Deployment**: The process of deploying the model to Edge Impulse using the TensorFlow Lite (TFLite) interpreter was attempted. However, the simulation of inference using the Edge Impulse API encountered issues, as it doesn't directly support an inference() method for the model deployment. Instead, TFLite's interpreter was used to handle inference. This step required additional setup to ensure compatibility with the simulation environment.
* **Inference Simulation**: The simulation involved running inference across different edge devices. The execution time and accuracy were tracked, but issues were encountered when using non-PAC (Precision-Aware Computing) devices. These devices showed unpredictable performance due to hardware limitations on the TFLite interpreters. Additionally, inference was done on a single sample per run instead of using batched data, which would have better simulated real-world use cases.

## **2. Observations**

* **Incompatibility in Data Types**: A mismatch between the data types of input tensors (tf.float32) and the labels (int64) led to issues during inference. This was resolved by ensuring the input data type was consistent with the model's expectations.
* **Simulation Limitations**: The attempt to use Edge Impulse's inference() method directly caused runtime errors, as it was not supported for this type of model. Instead, the inference was handled by using the TFLite interpreter, but additional steps were needed to run the simulation successfully. Specifically, the code needed to ensure it was processing multiple samples in batches to reflect more realistic edge device behavior.
* **Performance Challenges**: Non-PAC devices presented difficulties in performance simulation. The hardware limitations of these devices caused inconsistent results, highlighting the need for specialized support or optimization for these types of devices. Additionally, the lack of batch processing during inference reduced the accuracy and realism of the simulation.

## **3. Documentation of Results**

* **Accuracy**: The final accuracy of the model, after simulating inference across different edge devices, was approximately 97.26%. This result was based on running the model with a single sample for each device. While this was sufficient for the simulation, real-world edge devices would benefit from batch processing to improve both performance and accuracy.
* **Limitations**: The simulation was limited by the use of Edge Impulse, which doesn't handle non-PAC devices or batched data as effectively as expected. This restriction impacted the accuracy and reliability of the results, especially for devices that don't support the PAC features or have limited computational power. Furthermore, using single samples for inference rather than batches was a simplified approach, which may not fully represent the capabilities or performance of real edge devices.

### **Fixes and Improvements**

* **Data Types**: Ensure that both input and label tensors are of the same type (e.g., both float32) to avoid compatibility issues during model inference.
* **Inference Handling**: Replace the unsupported inference() method with the appropriate steps to run inference using the TFLite interpreter. This may involve adding specific code to load the model and perform inference on batched data.
* **Device Simulation**: Consider optimizing the simulation to handle edge devices with varying hardware capabilities. Testing on a range of devices that do not support PAC or have limited precision would provide a more realistic simulation.
* **Batch Processing**: To better simulate real-world edge inference, modify the code to process multiple samples in batches. This would provide more accurate results and better mimic how the model would perform on actual hardware.

This reflective journal captures key aspects of the simulation, highlights both successes and areas needing improvement.

Images : 