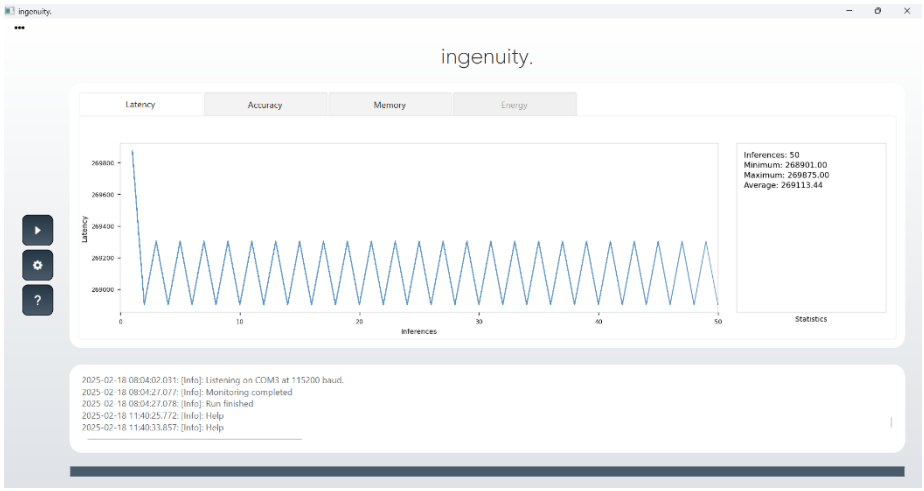
ingenuity.

Ingenuity is designed to benchmark the inference performance of ML models on embedded devices using its own inference engine.

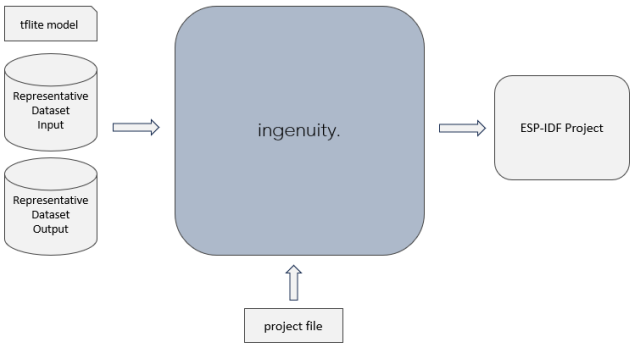
Benchmarking a quantized TFLite model typically involves multiple steps, including building and deploying the model on the device, as well as designing and implementing benchmarking test suites. Ingenuity automates this entire process with a single click, seamlessly bridging the gap between model quantization and benchmarking.



Through the Graphical User Interface (GUI), benchmark metrics such as inference latency, memory usage, and quantization accuracy can be easily monitored within seconds. This allows users to benchmark their models quickly and efficiently.

One-Click Process

Before execution, the project file must be properly configured with the validator's input and output representative datasets, as well as inference settings such as the inference rate and the number of inferences for benchmarking. Once configured, a single click automates the entire process—handling file generation, project building, flashing, and real-time monitoring of benchmarking results.



After the benchmark is completed, the generated ESP-IDF project folder can be used to integrate the benchmarking setup with the user’s application code.

Getting-started

The **Home** button allows you to create a new project or load an existing one. To create a new project, follow these steps:

1. Click the **Home** button and select **"New Project..."**
2. In the **New Project** window, choose the folder where the project will be created.
3. Enter a name for the new project and click **OK**—this will open a file explorer window.
4. Locate and open the **.yaml** file, then edit it with the appropriate parameters.

```yaml
# Configuration file for the Ingenuity software.
# Note: If relative paths are used, they will be considered relative to this YAML file.
#       The 'toolchain_path' must be an absolute path.
#       The settings in this file can be modified later from the GUI.

main:
  output_directory: "esp32s3"               # Directory where the IDF project files will be stored
  model_file: "model.tflite"                # Path to the TensorFlow Lite model file

device:
  manufacturer: "Espressif"                 # Name of the device manufacturer
  dev_model: "ESP32-S3"                     # Specific development model being used
  toolchain_path: "C:/Espressif_5.3.1"      # Absolute path to the ESP-IDF toolchain

validator:
  input_dataset: "rep_dataset_input.csv"    # CSV file containing the representative dataset
  output_dataset: "rep_dataset_output.csv"  # CSV file containing the expected output of the representative dataset

settings:
  inference_rate: 10                        # Inference interval in milliseconds (valid range: 50 to 1000 ms)
  inferences_n: 100                         # Total number of inferences to run (valid range: 1 to 10^9)
  show_graphs: true                         # Whether to display performance graphs (true or false)
```

The **Execution** button starts the one-click benchmarking process. It becomes enabled after a project is loaded and consists of the following steps:

1. Generates the **ESP-IDF project**, including the **Ingenuity inference engine** library.
2. Creates the **main C file** and the **validator files** required for benchmarking.
3. Builds the project and flashes it to the device.
4. Monitors the device output and displays the benchmark results.

The **Settings** button opens the settings window, allowing you to configure benchmark parameters.

Note: These parameters can also be modified directly in the project file before loading the project.

The **main panel** displays benchmark results for the following metrics:

1. **Latency** – Inference latency measured in MCU cycles.
2. **Accuracy** – Accuracy of the inference engine, calculated by comparing the actual output with the representative output dataset.
3. **Memory** –
   - The first table shows the device's overall memory usage.
   - The second table shows the inference engine’s memory usage as a separate component.
4. **Energy** – This feature is currently under development.

Compatibility & Use Cases

Ingenuity supports fully quantized (int8) TensorFlow Lite ML models based on fully connected feed-forward neural networks. Its inference engine is optimized to utilize the AI hardware accelerators and internal memory of the ESP32-S3 microcontroller.