# Weekly Progress Report

## *Project: TinyML-Based Project on FPGA Board with RISC-V Core*

**Name**: Aman Chauhan
**Branch:** Btech. Computer Science and Engineering (CSE Core)
**Roll Number:** 22BCE0476
**College:** Vellore Institute of Technology, Vellore
**Mentor**: Dr. Sudip Roy

***Week:*** *3rd report*
***Date:*** *6th May 2025*

## RVFPGA TRACE USAGE

**1. Simulation-Specific SoC Configuration**

The rvfpgasim.v top module configures the SweRVolf SoC for tracing by:

•Instantiating the core swervolf_core with simulation-only memory interfaces17

•Disabling ViDBo and Pipeline features via parameter settings (ViDBo=0, Pipeline=0)17

•Exposing internal buses and control signals to Verilator's tracing infrastructure

**2. Cyclic Signal Capture Mechanism**

The tb.cpp driver implements a precise simulation loop that:

•Toggles the main clock every iteration (half-cycle resolution)59

•Triggers Verilator's VCD tracing on each positive clock edge58

•Captures all hierarchical signals through trace->dump() calls910

**3. Interactive Waveform Analysis**

GTKWave enables deep inspection of the trace.vcd through:

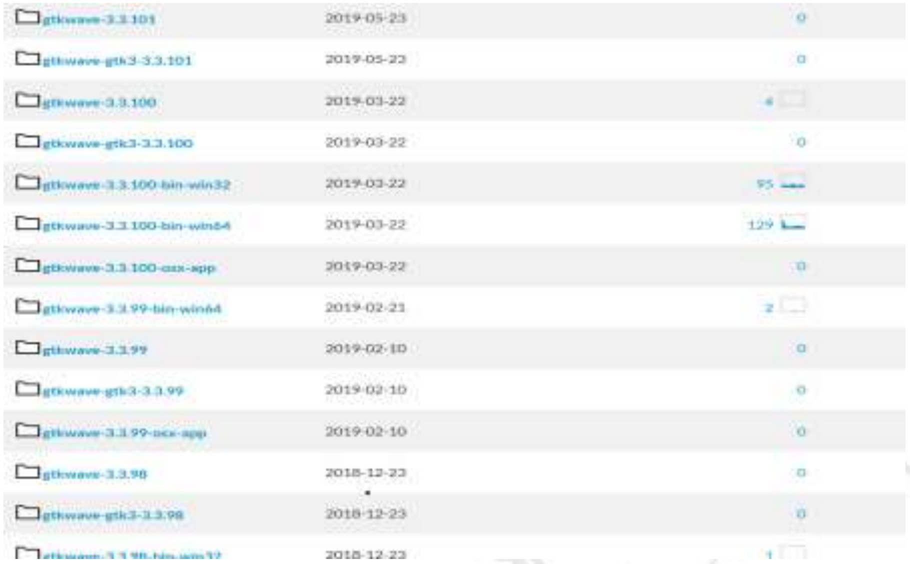
•Hierarchical signal grouping using preconfigured TCL scripts69

•Time-aligned views of pipeline stages and peripheral registers710

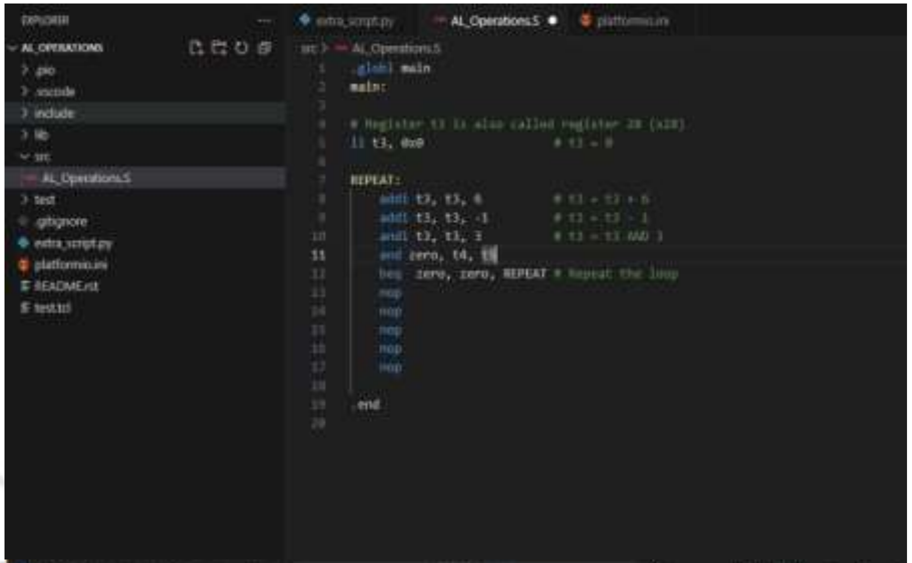
•Custom value translators for RISC-V instructions and CSRs10

IIT ROORKEE

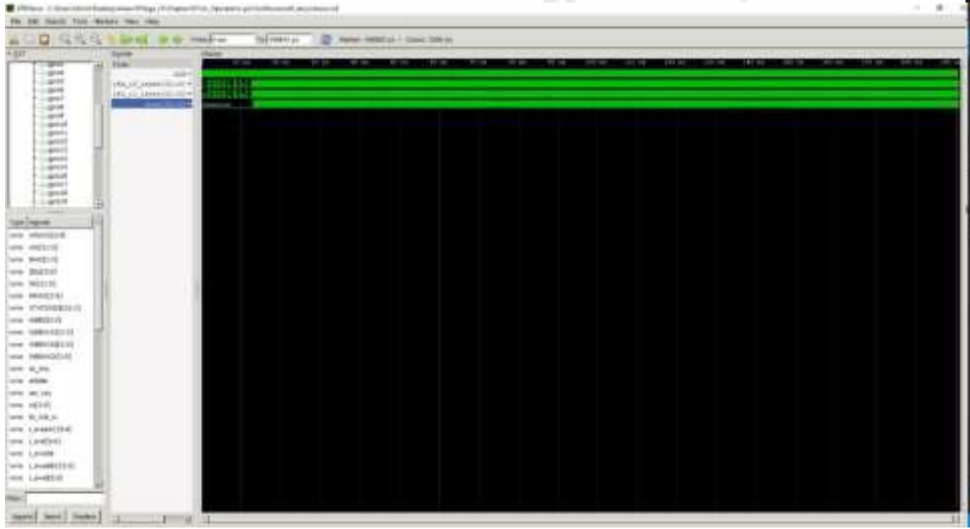
# RVFPGA TRACE

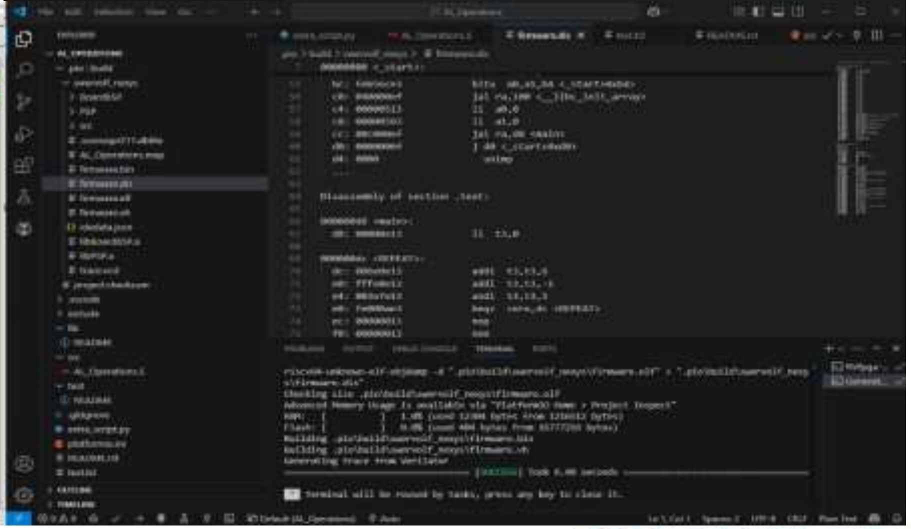


| | | |
|---|---|---|
| gtkwave-3.3.101 | 2019-05-23 | 0 |
| gtkwave-gtk3-3.3.101 | 2019-05-22 | 0 |
| gtkwave-3.3.100 | 2019-03-22 | 4 |
| gtkwave-gtk3-3.3.100 | 2019-03-22 | 0 |
| gtkwave-3.3.100-bin-win32 | 2019-03-22 | 55 |
| gtkwave-3.3.100-bin-win64 | 2019-03-22 | 129 |
| gtkwave-3.3.100-osx-app | 2019-03-22 | 0 |
| gtkwave-3.3.99-bin-win64 | 2019-02-21 | 2 |
| gtkwave-3.3.99 | 2019-02-10 | 0 |
| gtkwave-gtk3-3.3.99 | 2019-02-10 | 0 |
| gtkwave-3.3.99-osx-app | 2019-02-10 | 0 |
| gtkwave-3.3.98 | 2018-12-23 | 0 |
| gtkwave-gtk3-3.3.98 | 2018-12-23 | 0 |
| gtkwave-3.3.98-bin-win32 | 2018-12-23 | 1 |



```
.globl main
main:

# Register t3 is also called register 28 (x28)
li t3, 0x0                  # t3 = 0

REPEAT:
    addi t3, t3, 6          # t3 = t3 + 6
    addi t3, t3, -1         # t3 = t3 - 1
    andi t3, t3, 3          # t3 = t3 AND 3
    and zero, t4, t4
    beq  zero, zero, REPEAT # Repeat the loop
    nop
    nop
    nop
    nop
    nop

.end
```

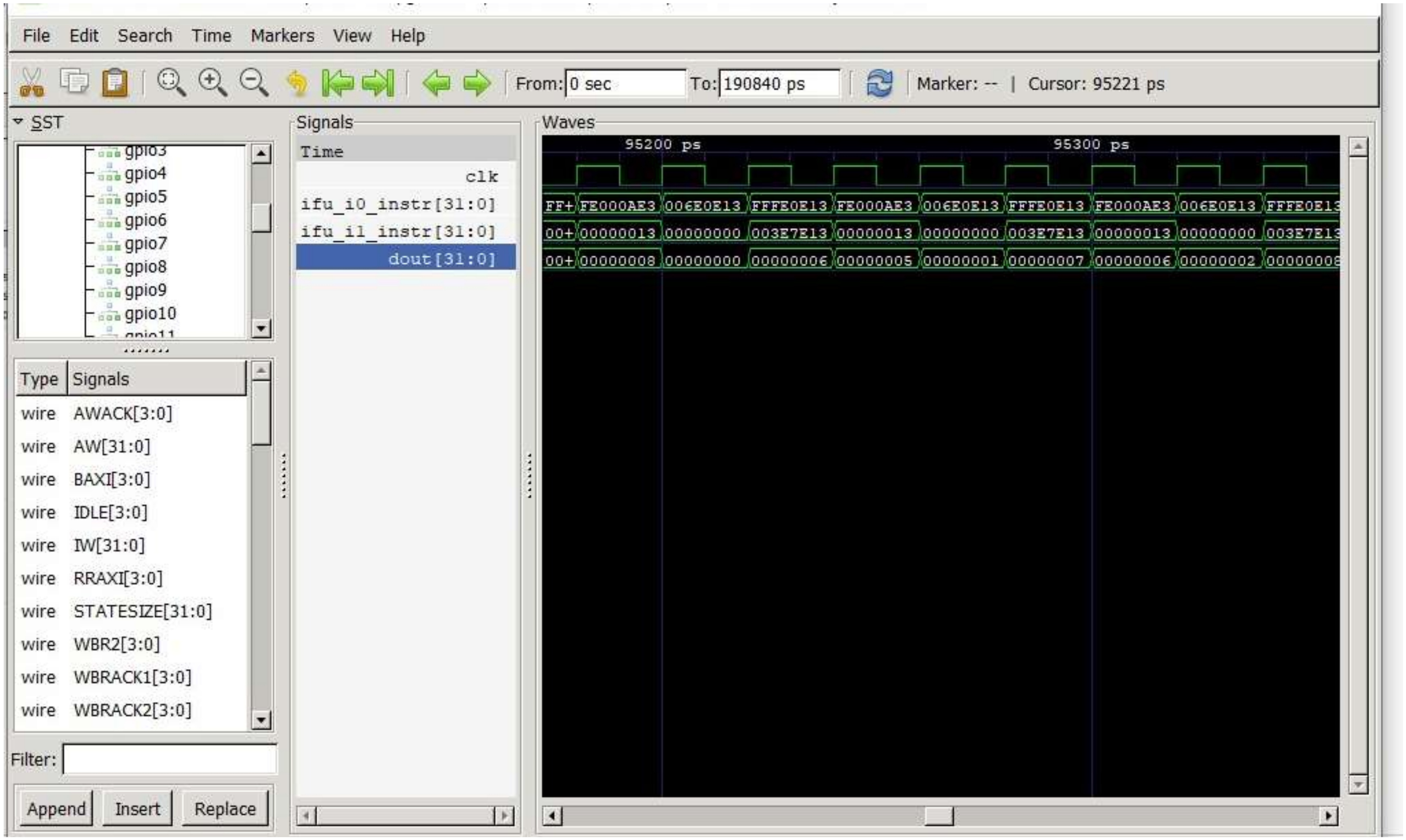




```
00000000 <_start>:
   [illegible]                jal ra,[illegible] <__libc_init_array>
   [illegible]                li  a0,0
   [illegible]                li  a1,0
   [illegible]                jal ra,d8 <main>
   [illegible]                j [illegible] <_start+[illegible]>
   [illegible]                unimp
   ...

Disassembly of section .text:

000000d8 <main>:
   d8: [illegible]            li  t3,0

000000dc <REPEAT>:
   dc: [illegible]            addi  t3,t3,6
   e0: fff[illegible]         addi  t3,t3,-1
   e4: [illegible]            andi  t3,t3,3
   e8: [illegible]            beqz  zero,dc <REPEAT>
   ec: 00000013               nop
   f0: 00000013               nop
```

```
riscv64-unknown-elf-objdump -d ".pio\build\swervolf_nexys\firmware.elf" > ".pio\build\swervolf_nexys\firmware.dis"
Checking size .pio\build\swervolf_nexys\firmware.elf
Advanced Memory Usage is available via "PlatformIO Home > Project Inspect"
RAM:   [          ]   1.0% (used 12308 bytes from 1210012 bytes)
Flash: [          ]   0.0% (used 404 bytes from 33777250 bytes)
Building .pio\build\swervolf_nexys\firmware.bin
Building .pio\build\swervolf_nexys\firmware.vh
Generating trace from Verilator
==================== [SUCCESS] Took 0.40 seconds ====================

 *  Terminal will be reused by tasks, press any key to close it.
```

## GTK Wave WIN62bit version



| Type | Signals |
|---|---|
| wire | AWACK[3:0] |
| wire | AW[31:0] |
| wire | BAXI[3:0] |
| wire | IDLE[3:0] |
| wire | IW[31:0] |
| wire | RRAXI[3:0] |
| wire | STATESIZE[31:0] |
| wire | WBR2[3:0] |
| wire | WBRACK1[3:0] |
| wire | WBRACK2[3:0] |

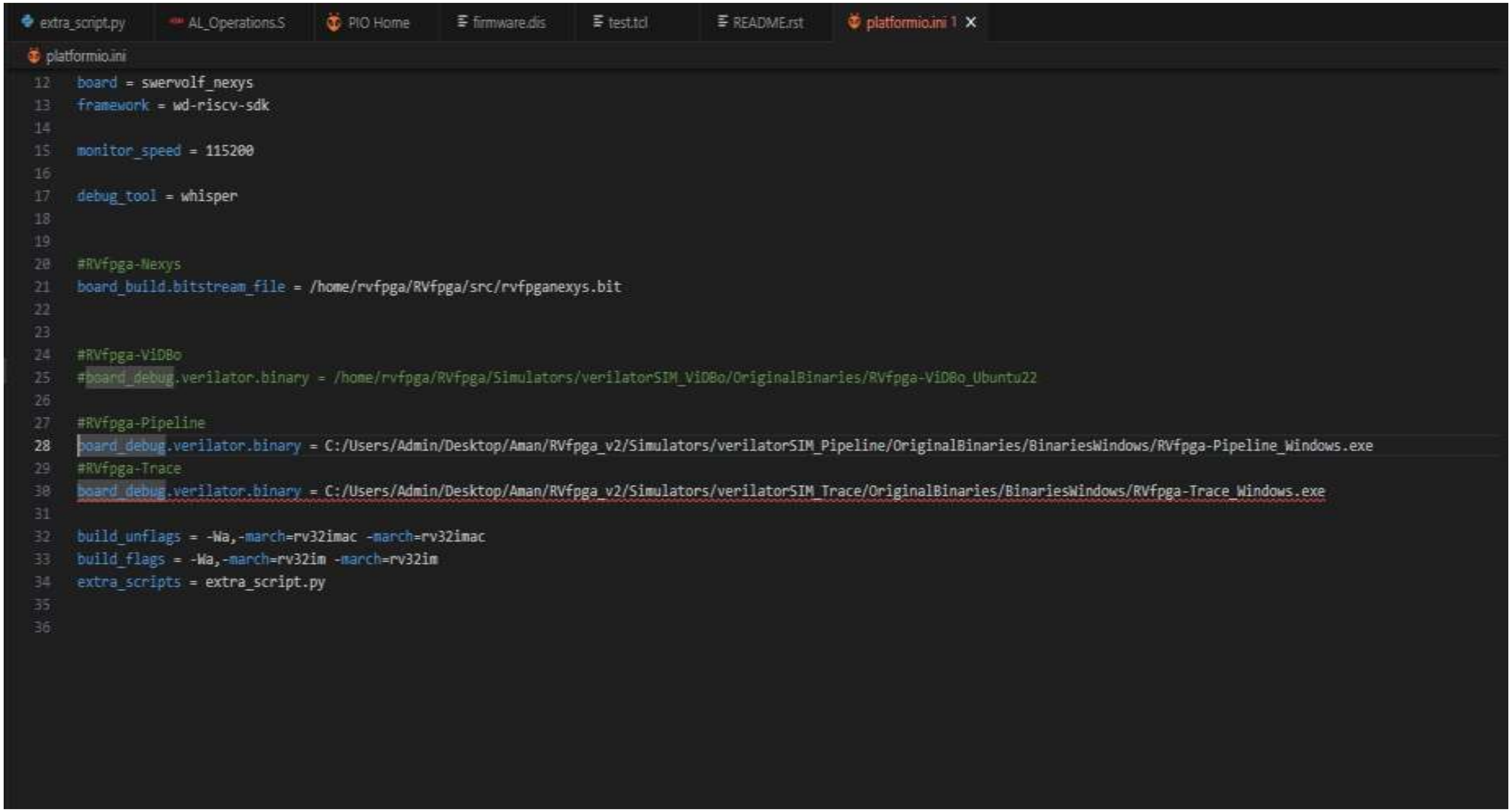
```
Time                          95200 ps                                         95300 ps
clk
ifu_i0_instr[31:0]  FF+ FE000AE3 006E0E13 FFFE0E13 FE000AE3 006E0E13 FFFE0E13 FE000AE3 006E0E13 FFFE0E13
ifu_i1_instr[31:0]  00+ 00000013 00000000 003E7E13 00000013 00000000 003E7E13 00000013 00000000 003E7E13
dout[31:0]          00+ 00000008 00000000 00000006 00000005 00000001 00000007 00000006 00000002 00000008
```

## RVFPGA whisper Usage

1**. Functional Simulation Without Hardware**
Whisper allows users to run and debug RISC-V assembly and C code entirely in software, simulating the behavior of the SweRV EH1 core without requiring physical hardware or an FPGA351.

2. **Interactive Debugging and Inspection**
Users can single-step through code, inspect and modify RISC-V registers, and access simulated system memory. This interactive mode is especially useful for debugging and understanding program execution at the instruction level152.

3. **Golden Model for Verification**
Whisper can run in lock-step with a Verilog simulator, serving as a reference ("golden model") to verify the correctness of hardware implementations after each instruction of a test program153. This is valuable for ensuring that hardware designs faithfully execute intended software behavior

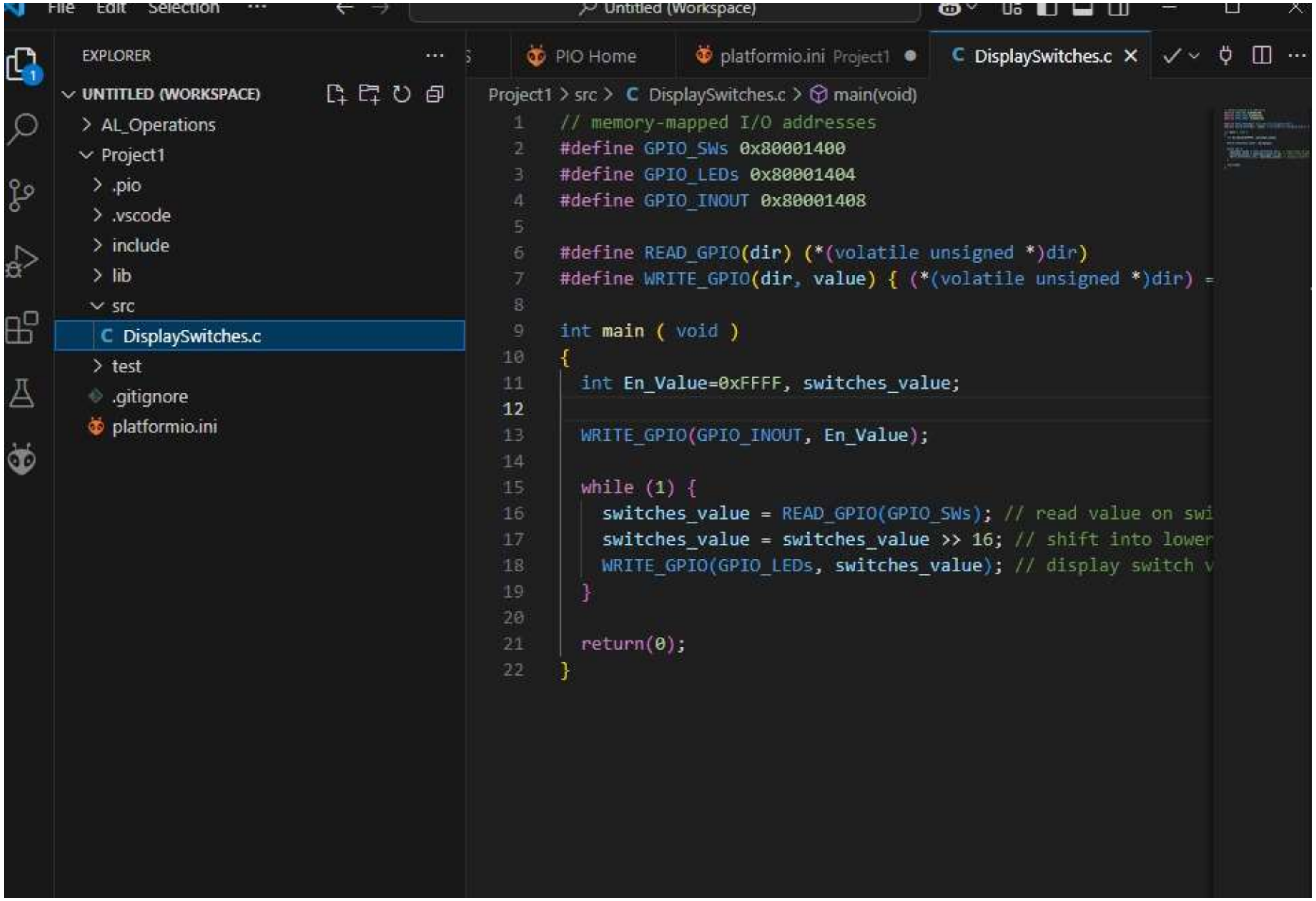

extra_script.py | AL_Operations.S | PIO Home | firmware.dis | test.tcl | README.rst | platformio.ini 1

platformio.ini

```
board = swervolf_nexys
framework = wd-riscv-sdk

monitor_speed = 115200

debug_tool = whisper


#RVfpga-Nexys
board_build.bitstream_file = /home/rvfpga/RVfpga/src/rvfpganexys.bit


#RVfpga-ViDBo
#board_debug.verilator.binary = /home/rvfpga/RVfpga/Simulators/verilatorSIM_ViDBo/OriginalBinaries/RVfpga-ViDBo_Ubuntu22

#RVfpga-Pipeline
board_debug.verilator.binary = C:/Users/Admin/Desktop/Aman/RVfpga_v2/Simulators/verilatorSIM_Pipeline/OriginalBinaries/BinariesWindows/RVfpga-Pipeline_Windows.exe
#RVfpga-Trace
board_debug.verilator.binary = C:/Users/Admin/Desktop/Aman/RVfpga_v2/Simulators/verilatorSIM_Trace/OriginalBinaries/BinariesWindows/RVfpga-Trace_Windows.exe

build_unflags = -Wa,-march=rv32imac -march=rv32imac
build_flags = -Wa,-march=rv32im -march=rv32im
extra_scripts = extra_script.py


```

# Compiler C program Using whisper

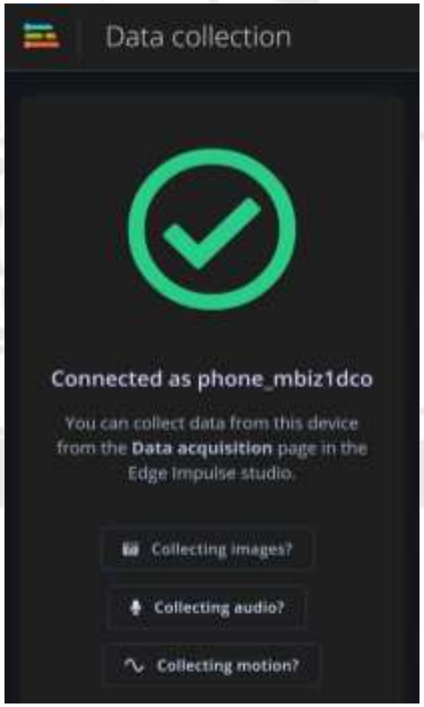
- 1. **Functional Simulation Without Hardware**
  Whisper is a RISC-V instruction set simulator that allows you to run and debug RISC-V assembly and C programs entirely in software, without requiring physical hardware or an FPGA. This is ideal for verifying program logic and debugging before deploying to hardware124.
- 2. **Interactive Debugging Capabilities**
  Whisper provides an interactive mode where users can single-step through code, inspect and modify registers, and examine simulated system memory. This makes it a powerful tool for learning, development, and troubleshooting at the instruction level125.
- 3. **Integration with RVfpga Workflow**
  Whisper is integrated into the RVfpga toolchain and can be used alongside other simulation tools (such as RVfpga-Trace and RVfpga-Pipeline) to provide a comprehensive software simulation environment. This enables users to complete all RVfpga labs and experiments in simulation, even if they do not have access to an FPGA board345.



```c
// memory-mapped I/O addresses
#define GPIO_SWs 0x80001400
#define GPIO_LEDs 0x80001404
#define GPIO_INOUT 0x80001408

#define READ_GPIO(dir) (*(volatile unsigned *)dir)
#define WRITE_GPIO(dir, value) { (*(volatile unsigned *)dir) =

int main ( void )
{
  int En_Value=0xFFFF, switches_value;

  WRITE_GPIO(GPIO_INOUT, En_Value);

  while (1) {
    switches_value = READ_GPIO(GPIO_SWs); // read value on swi
    switches_value = switches_value >> 16; // shift into lower
    WRITE_GPIO(GPIO_LEDs, switches_value); // display switch v
  }

  return(0);
}
```

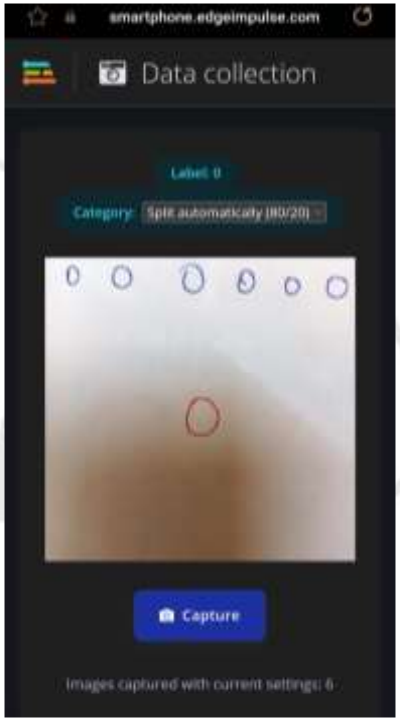
## Tiny ML model

- **Objective**: To classify images of digits 0 and 1 using a TinyML-compatible model.
- **Tool Used**: Edge Impulse
- **Approach**: Transfer Learning for image classification.



Connected as phone_mbiz1dco

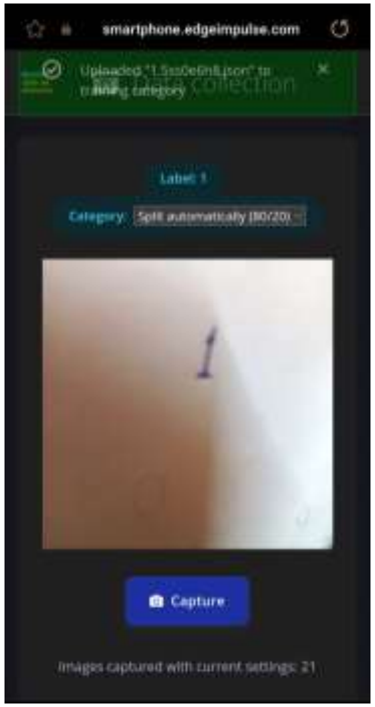
You can collect data from this device from the **Data acquisition** page in the Edge Impulse studio.



Label: 0

Category: Split automatically (80/20)

Images captured with current settings: 6



Uploaded "1.5ss0e6n8.json" to training category

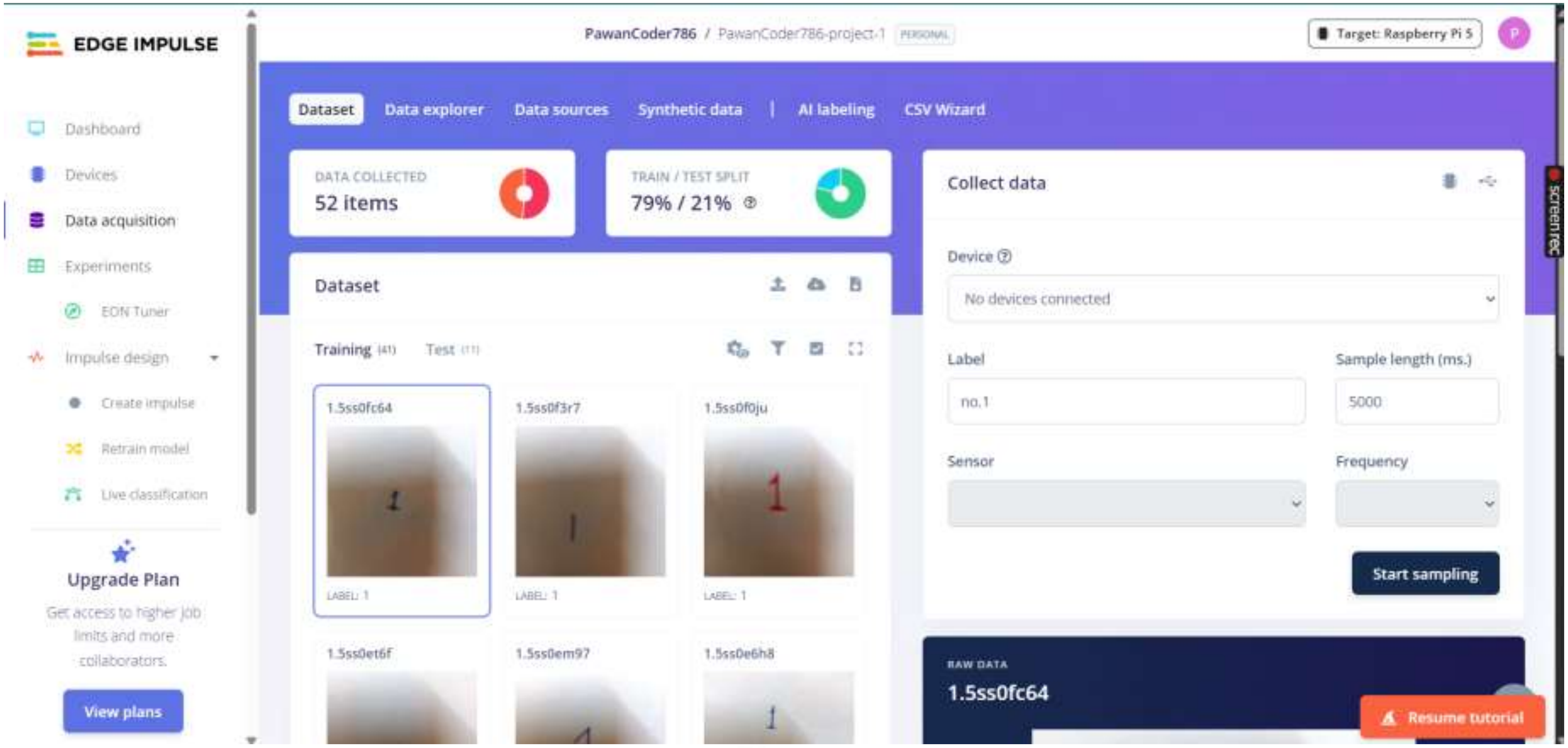
Label: 1

Category: Split automatically (80/20)

Images captured with current settings: 21

**Fig: Collection of the dataset**(through mobile phone as secondary device)

# Dataset Collected at Edge Impulse



PawanCoder786 / PawanCoder786-project-1 PERSONAL

Target: Raspberry Pi 5

Get access to higher job limits and more collaborators.

DATA COLLECTED
52 items

TRAIN / TEST SPLIT
79% / 21%

Dataset

Training (41) Test (11)

| | | |
|---|---|---|
| 1.5ss0fc64 LABEL: 1 | 1.5ss0f3r7 LABEL: 1 | 1.5ss0f0ju LABEL: 1 |
| 1.5ss0et6f | 1.5ss0em97 | 1.5ss0e6h8 |

Collect data

Device: No devices connected

Label: no.1

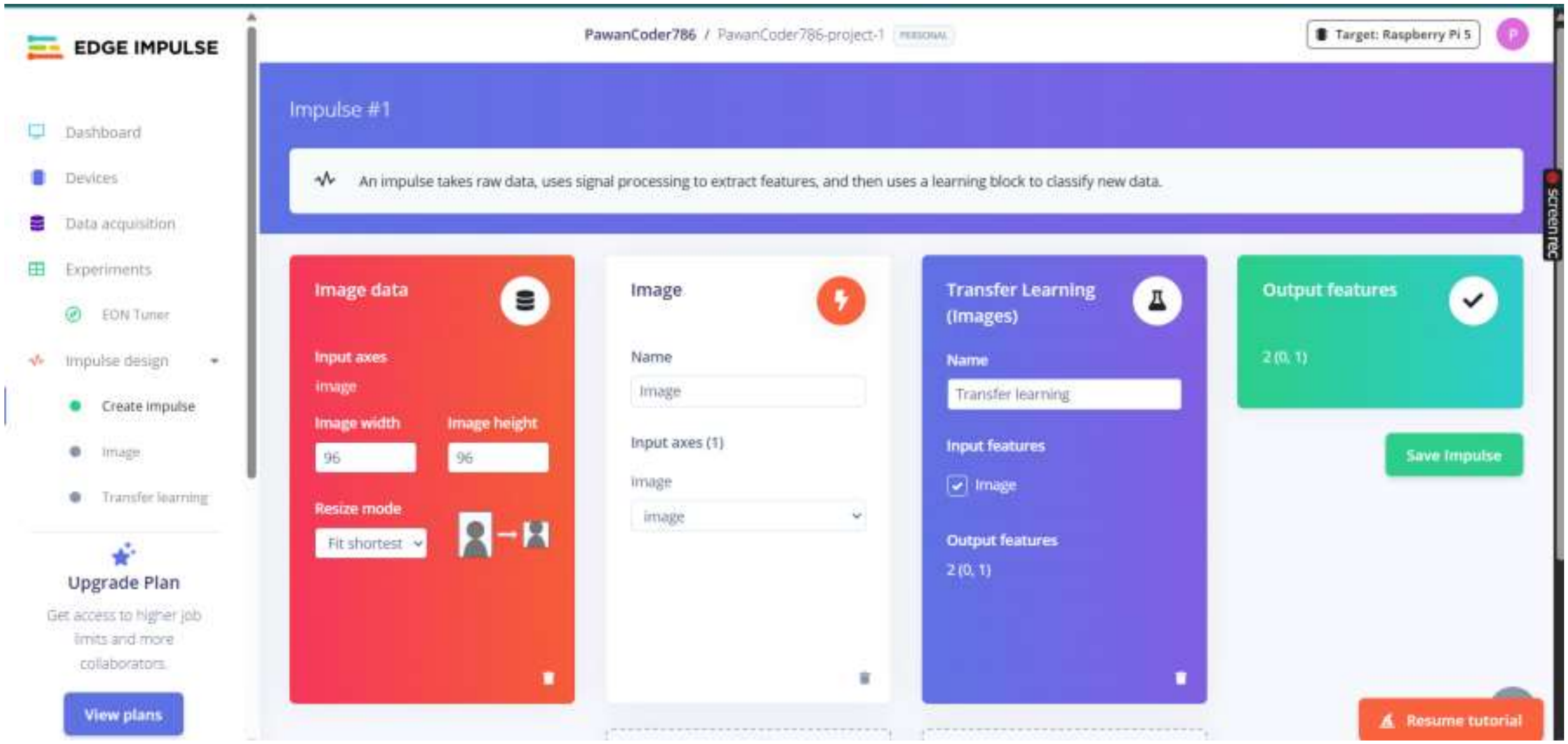
Sample length (ms.): 5000

Sensor

Frequency

RAW DATA
1.5ss0fc64

# Impulse design (create impulse)



PawanCoder786 / PawanCoder786-project-1 PERSONAL

Target: Raspberry Pi 5

Impulse #1

An impulse takes raw data, uses signal processing to extract features, and then uses a learning block to classify new data.

**Image data**

Input axes: image

Image width: 96

Image height: 96

Resize mode: Fit shortest

**Image**

Name: Image

Input axes (1): image

**Transfer Learning (Images)**

Name: Transfer learning

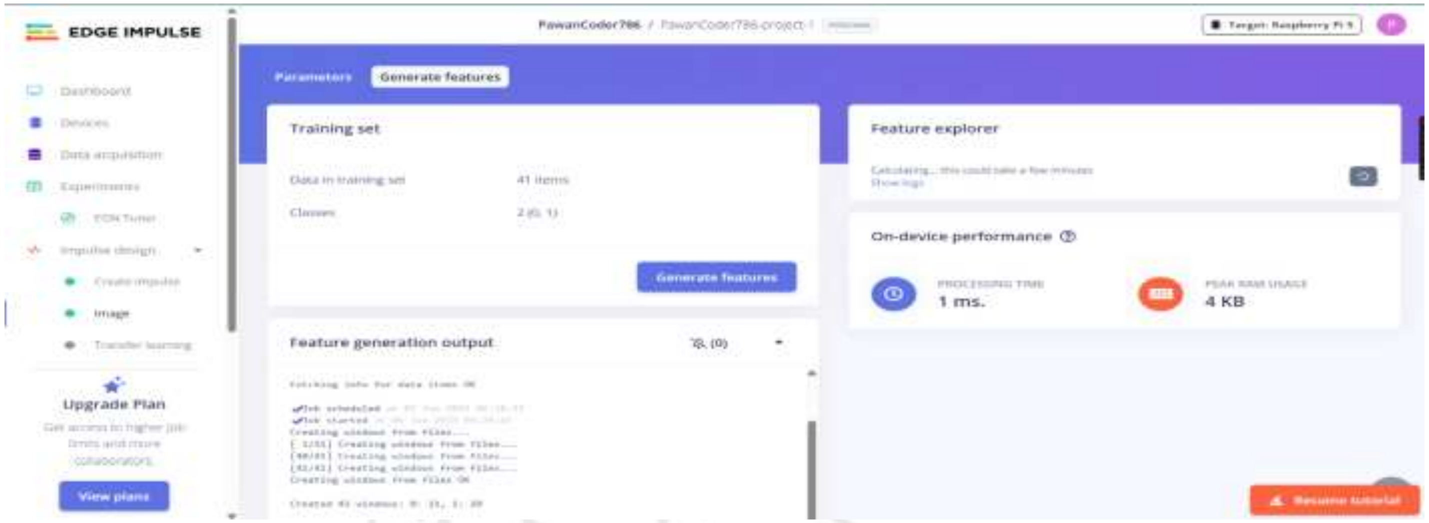
Input features: ✓ Image

Output features: 2 (0, 1)

**Output features**

2 (0, 1)

# Feature Explorer (Data Visualization)



Training set

| | |
|---|---|
| Data in training set | 41 items |
| Classes | 2 (0, 1) |

Generate features

Get access to higher job limits and more collaborators.

Feature generation output

```
Fetching info for data items OK

✓Job scheduled at [illegible]
✓Job started at [illegible]
Creating windows from files...
[ 1/41] Creating windows from files...
[40/41] Creating windows from files...
[41/41] Creating windows from files...
Creating windows from files OK

Created 41 windows: 0: 21, 1: 20
```

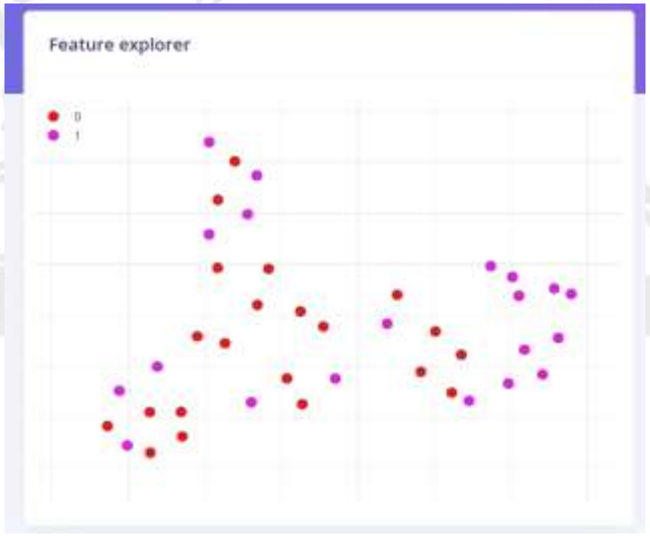
Feature explorer

Calculating... this could take a few minutes
Show logs

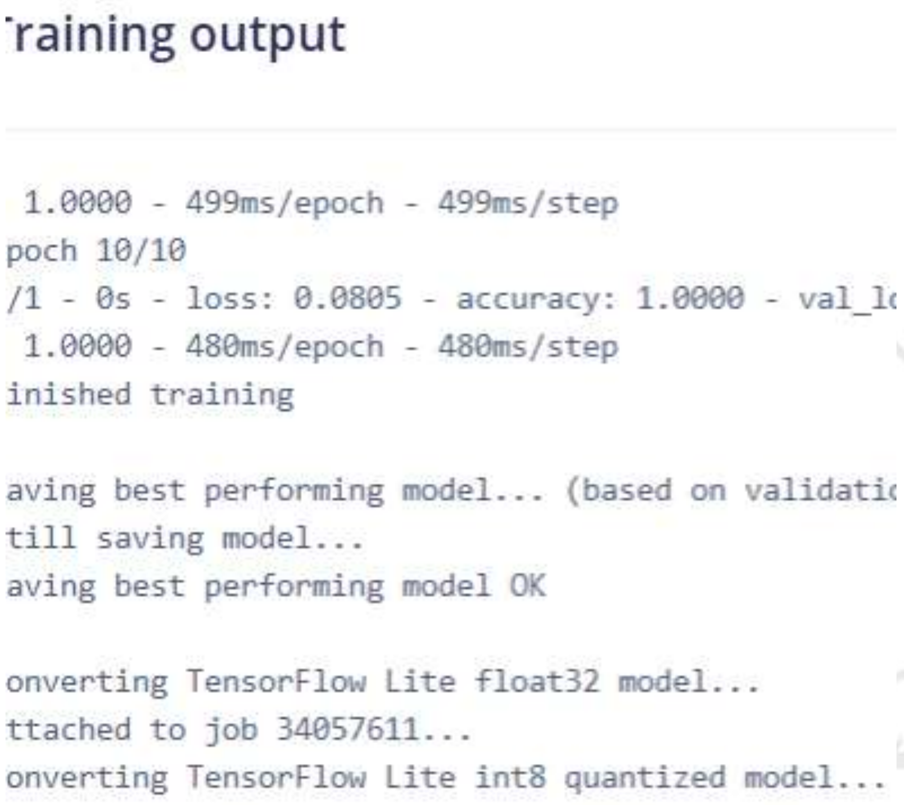
On-device performance

PROCESSING TIME 1 ms.

PEAK RAM USAGE 4 KB

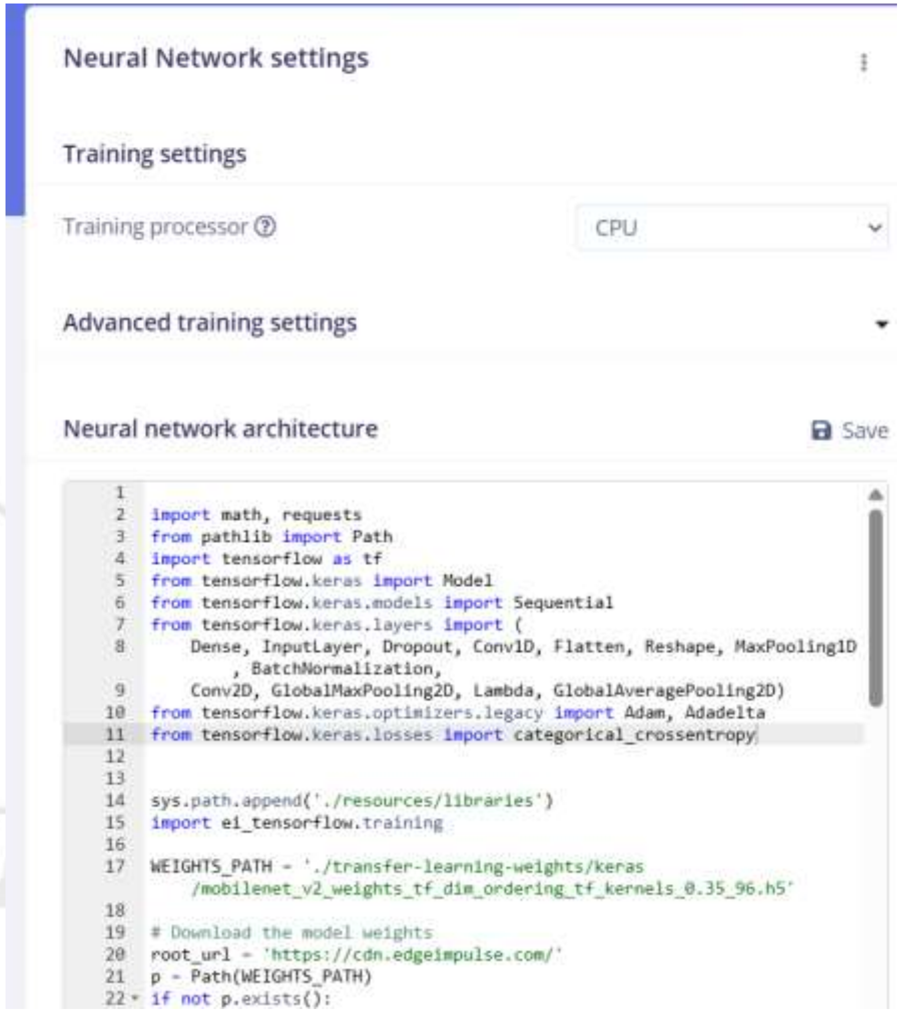


## Training of dataset(for 20 epochs,dataset-56 images)



'raining output

```
 1.0000 - 499ms/epoch - 499ms/step
poch 10/10
/1 - 0s - loss: 0.0805 - accuracy: 1.0000 - val_lo
 1.0000 - 480ms/epoch - 480ms/step
inished training

aving best performing model... (based on validatio
till saving model...
aving best performing model OK

onverting TensorFlow Lite float32 model...
ttached to job 34057611...
onverting TensorFlow Lite int8 quantized model...
```



Neural Network settings

Training settings

Training processor: CPU

Advanced training settings

Neural network architecture

```python

import math, requests
from pathlib import Path
import tensorflow as tf
from tensorflow.keras import Model
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import (
    Dense, InputLayer, Dropout, Conv1D, Flatten, Reshape, MaxPooling1D
        , BatchNormalization,
    Conv2D, GlobalMaxPooling2D, Lambda, GlobalAveragePooling2D)
from tensorflow.keras.optimizers.legacy import Adam, Adadelta
from tensorflow.keras.losses import categorical_crossentropy


sys.path.append('./resources/libraries')
import ei_tensorflow.training

WEIGHTS_PATH = './transfer-learning-weights/keras
    /mobilenet_v2_weights_tf_dim_ordering_tf_kernels_0.35_96.h5'

# Download the model weights
root_url = 'https://cdn.edgeimpulse.com/'
p = Path(WEIGHTS_PATH)
if not p.exists():
```

## Google Collab Code(Main parameters)

```
INPUT_SHAPE = (96, 96, 3)

# Load the base model
base_model = tf.keras.applications.MobileNetV2(
   input_shape=INPUT_SHAPE,
   alpha=0.35,
   weights=WEIGHTS_PATH
)
base_model.trainable = False

# Build the model
model = Sequential()
model.add(InputLayer(input_shape=INPUT_SHAPE, name='x_input'))
last_layer_index = -3
model.add(Model(inputs=base_model.inputs,
outputs=base_model.layers[last_layer_index].output))
model.add(Reshape((-1, model.layers[-1].output.shape[3])))
model.add(Dense(16, activation='relu'))
model.add(Dropout(0.1))
model.add(Flatten())
model.add(Dense(classes, activation='softmax'))
```

```
# Training configuration
BATCH_SIZE = args.batch_size or 32
EPOCHS = args.epochs or 20
LEARNING_RATE = args.learning_rate or 0.0005
ENSURE_DETERMINISM = args.ensure_determinism
if not ENSURE_DETERMINISM:
    train_dataset = train_dataset.shuffle(buffer_size=BATCH_SIZE * 4)
prefetch_policy = 1 if ENSURE_DETERMINISM else tf.data.AUTOTUNE
train_dataset = train_dataset.batch(BATCH_SIZE, drop_remainder=False).prefetch(prefetch_policy)
validation_dataset = validation_dataset.batch(BATCH_SIZE, drop_remainder=False).prefetch(prefetch_policy)
callbacks.append(
    BatchLoggerCallback(BATCH_SIZE, train_sample_count, epochs=EPOCHS, ensure_determinism=ENSURE_DETERMINISM)
)
# Compile and train model
model.compile(
    optimizer=tf.keras.optimizers.Adam(learning_rate=LEARNING_RATE),
    loss='categorical_crossentropy',
    metrics=['accuracy']
)
model.fit(
    train_dataset,
    validation_data=validation_dataset,
    epochs=EPOCHS,
    verbose=2,
    callbacks=callbacks
)
```

•print('\nInitial training done.\n')

•# Fine-tuning configuration
•FINE_TUNE_EPOCHS = 10
•FINE_TUNE_PERCENTAGE = 65
•print(f'Fine-tuning best model for {FINE_TUNE_EPOCHS} epochs...\n')

**•# Load best model from initial training**
•model = ei_tensorflow.training.load_best_model(BEST_MODEL_PATH)

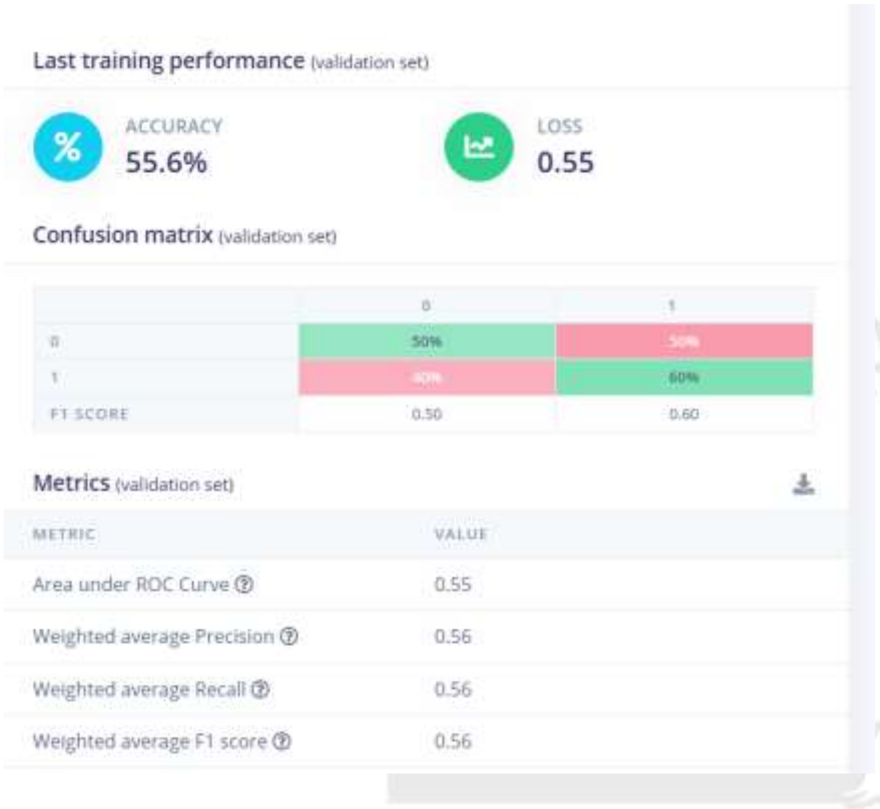
**•# Calculate fine-tuning start layer**
•model_layer_count = len(model.layers)
•fine_tune_from = math.ceil(model_layer_count * ((100 - FINE_TUNE_PERCENTAGE) / 100))

**•# Enable training on base model**
•model.trainable = True
•for layer in model.layers[:fine_tune_from]:
•    layer.trainable = False

**•# Compile and train (fine-tune)**
•model.compile(
•    optimizer=tf.keras.optimizers.Adam(learning_rate=0.000045),
•    loss='categorical_crossentropy',
•    metrics=['accuracy']
•)

•model.fit(
•    train_dataset,
•    epochs=FINE_TUNE_EPOCHS,
•    verbose=2,
•    validation_data=validation_dataset,
•    callbacks=callbacks,
•    class_weight=None
•)

## Model Performance



**Last training performance** (validation set)

ACCURACY
55.6%

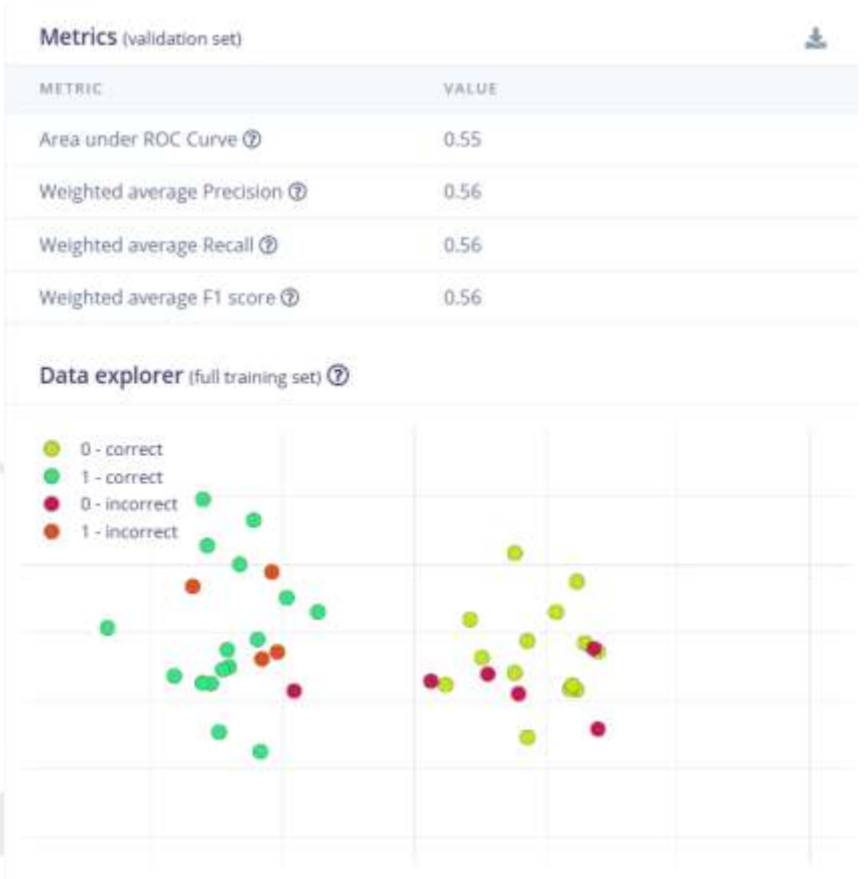
LOSS
0.55

**Confusion matrix** (validation set)

| | 0 | 1 |
|---|---|---|
| 0 | 50% | 50% |
| 1 | 40% | 60% |
| F1 SCORE | 0.50 | 0.60 |

**Metrics** (validation set)

| METRIC | VALUE |
|---|---|
| Area under ROC Curve | 0.55 |
| Weighted average Precision | 0.56 |
| Weighted average Recall | 0.56 |
| Weighted average F1 score | 0.56 |



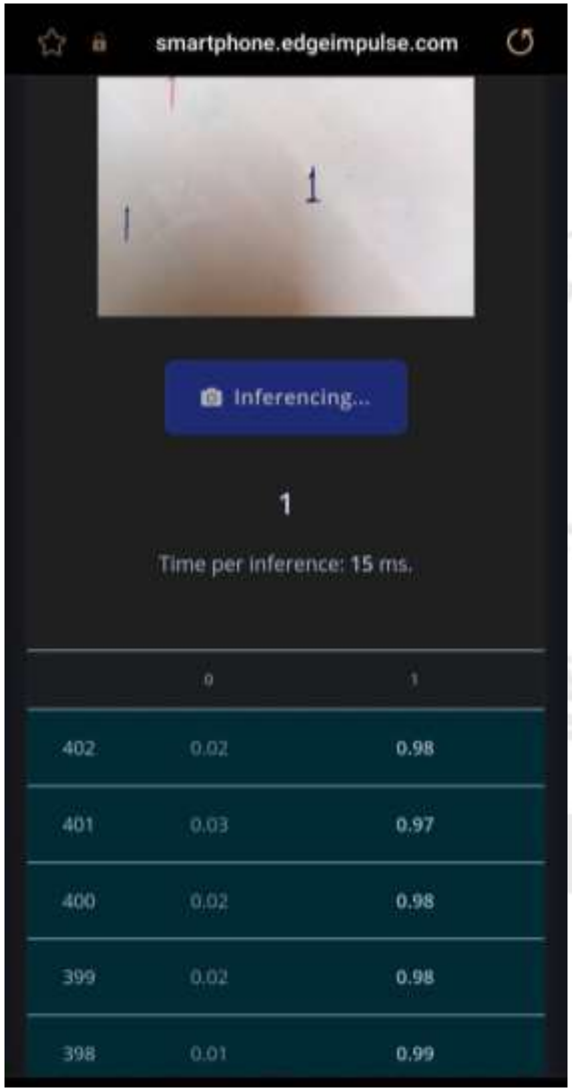
**Metrics** (validation set)

| METRIC | VALUE |
|---|---|
| Area under ROC Curve | 0.55 |
| Weighted average Precision | 0.56 |
| Weighted average Recall | 0.56 |
| Weighted average F1 score | 0.56 |

**Data explorer** (full training set)

**Not good accuracy due to less amount of dataset**

## Live Classification View

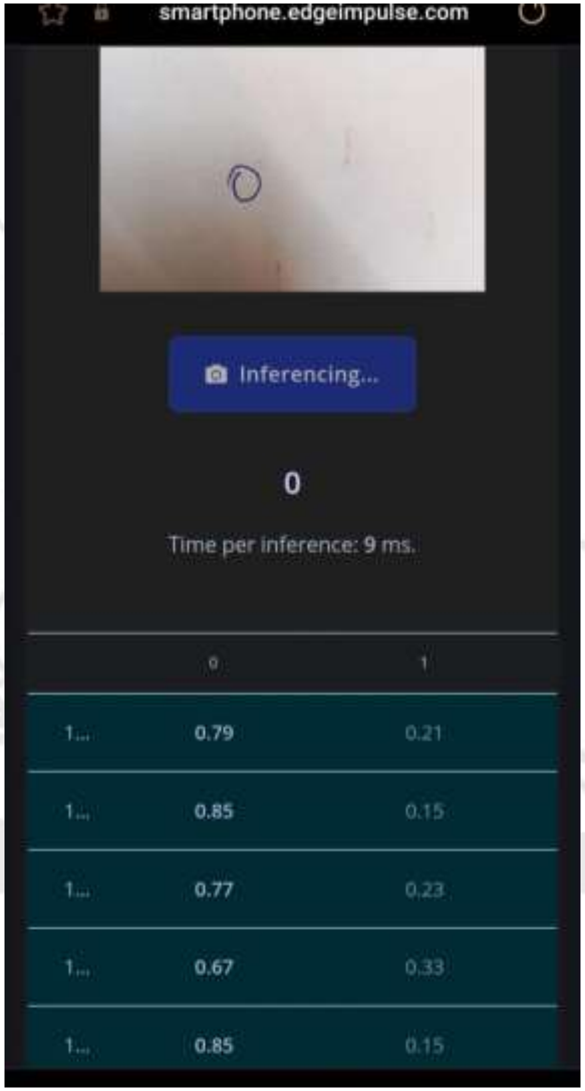

smartphone.edgeimpulse.com

Inferencing...

1

Time per inference: 15 ms.

| | 0 | 1 |
|---|---|---|
| 402 | 0.02 | 0.98 |
| 401 | 0.03 | 0.97 |
| 400 | 0.02 | 0.98 |
| 399 | 0.02 | 0.98 |
| 398 | 0.01 | 0.99 |

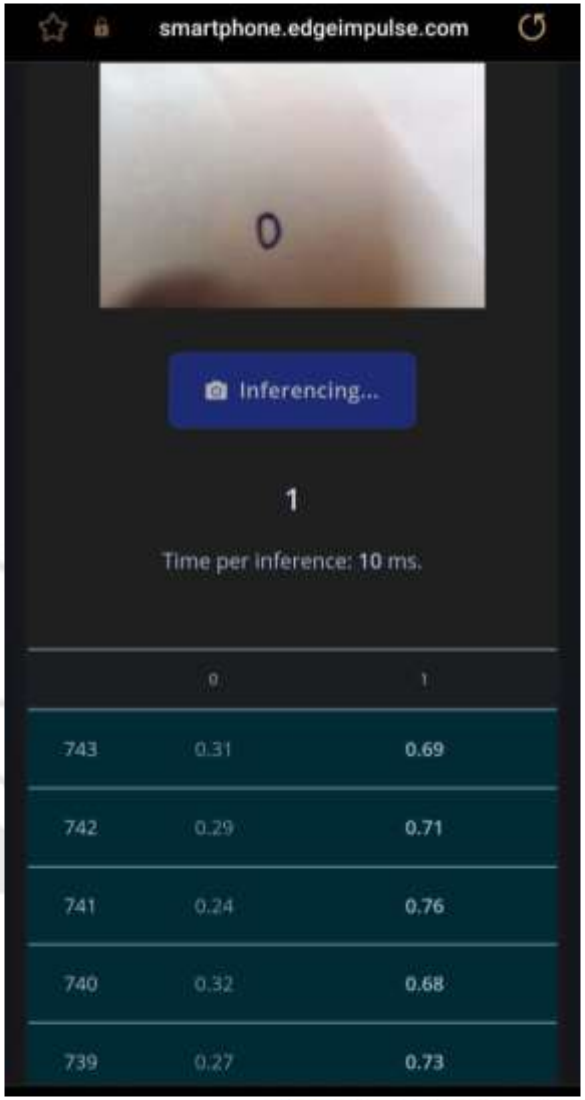

smartphone.edgeimpulse.com

Inferencing...

0

Time per inference: 9 ms.

| | 0 | 1 |
|---|---|---|
| 1... | 0.79 | 0.21 |
| 1... | 0.85 | 0.15 |
| 1... | 0.77 | 0.23 |
| 1... | 0.67 | 0.33 |
| 1... | 0.85 | 0.15 |


smartphone.edgeimpulse.com

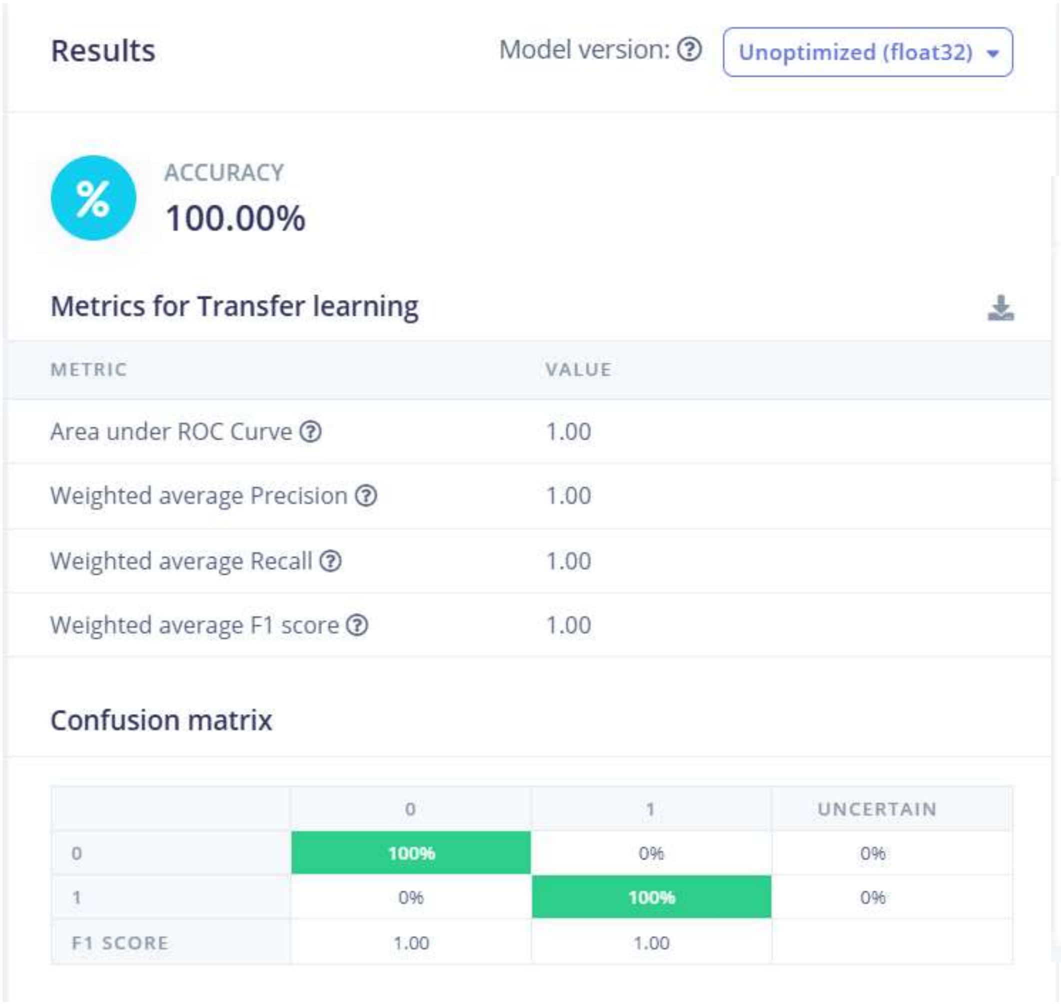
Inferencing...

1

Time per inference: 10 ms.

| | 0 | 1 |
|---|---|---|
| 743 | 0.31 | 0.69 |
| 742 | 0.29 | 0.71 |
| 741 | 0.24 | 0.76 |
| 740 | 0.32 | 0.68 |
| 739 | 0.27 | 0.73 |

# Testing of dataset



Results

Model version: ⓘ Unoptimized (float32)

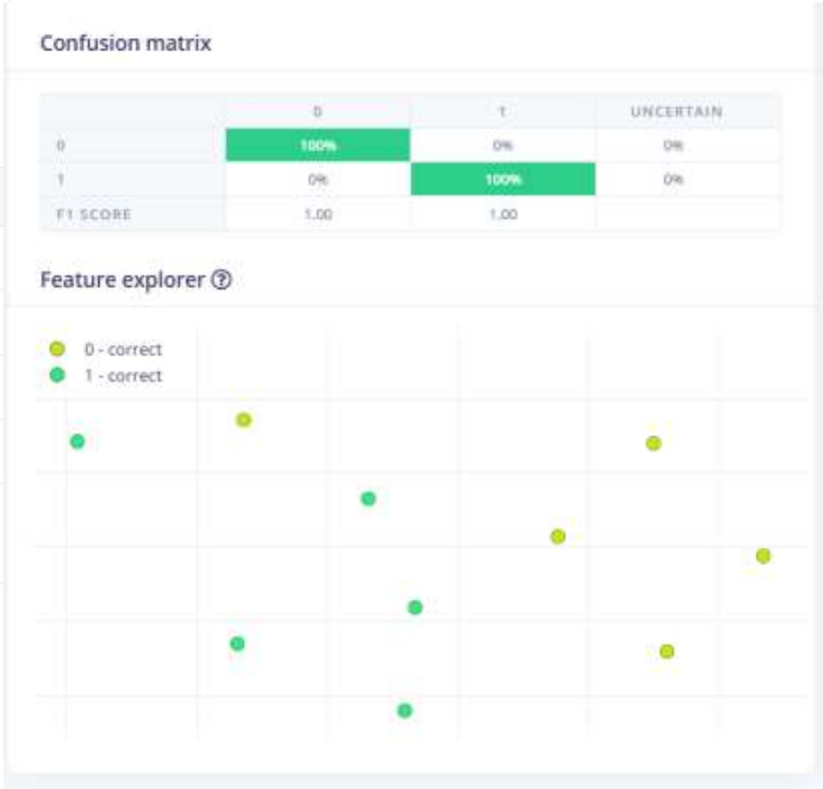
ACCURACY
100.00%

Metrics for Transfer learning

| METRIC | VALUE |
|---|---|
| Area under ROC Curve ⓘ | 1.00 |
| Weighted average Precision ⓘ | 1.00 |
| Weighted average Recall ⓘ | 1.00 |
| Weighted average F1 score ⓘ | 1.00 |

Confusion matrix

| | 0 | 1 | UNCERTAIN |
|---|---|---|---|
| 0 | 100% | 0% | 0% |
| 1 | 0% | 100% | 0% |
| F1 SCORE | 1.00 | 1.00 | |

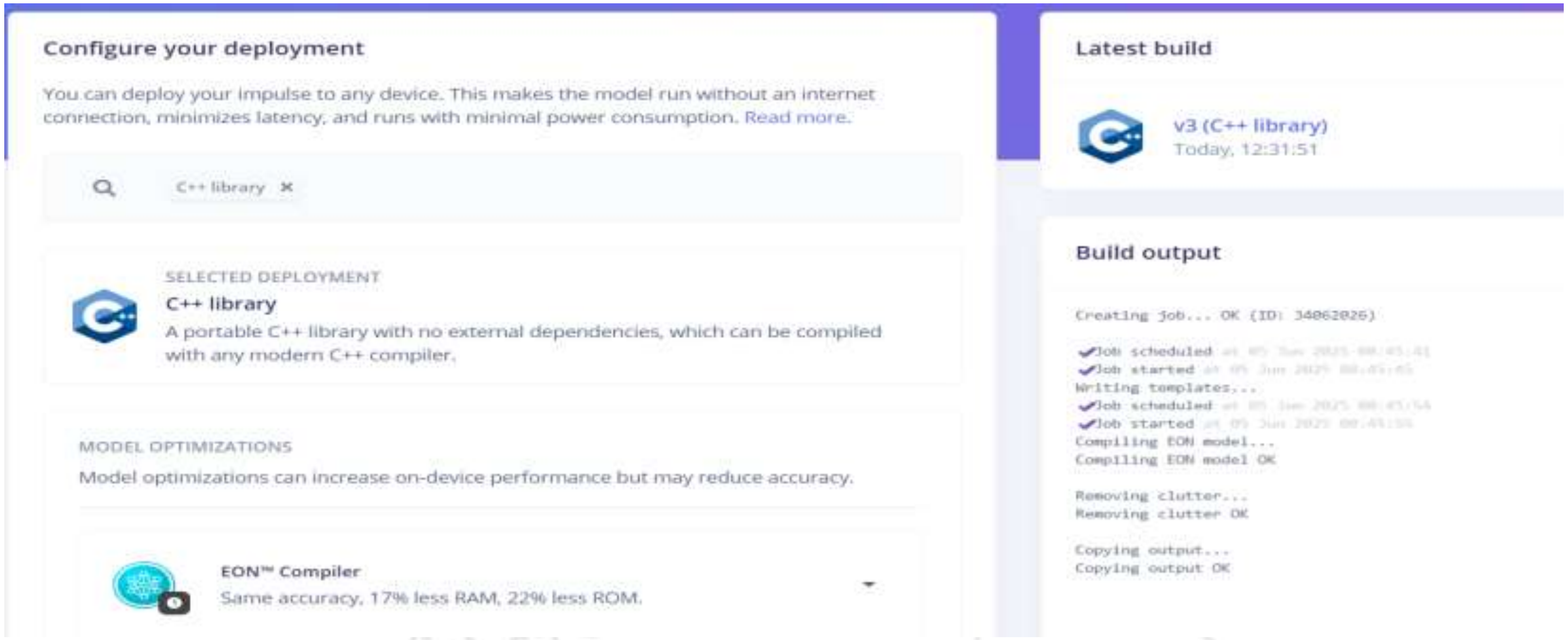


Confusion matrix

| | 0 | 1 | UNCERTAIN |
|---|---|---|---|
| 0 | 100% | 0% | 0% |
| 1 | 0% | 100% | 0% |
| F1 SCORE | 1.00 | 1.00 | |

Feature explorer ⓘ

# Deployment (Build TFLITE MODEL)



**Configure your deployment**

You can deploy your impulse to any device. This makes the model run without an internet connection, minimizes latency, and runs with minimal power consumption. Read more.

SELECTED DEPLOYMENT
**C++ library**
A portable C++ library with no external dependencies, which can be compiled with any modern C++ compiler.

MODEL OPTIMIZATIONS
Model optimizations can increase on-device performance but may reduce accuracy.

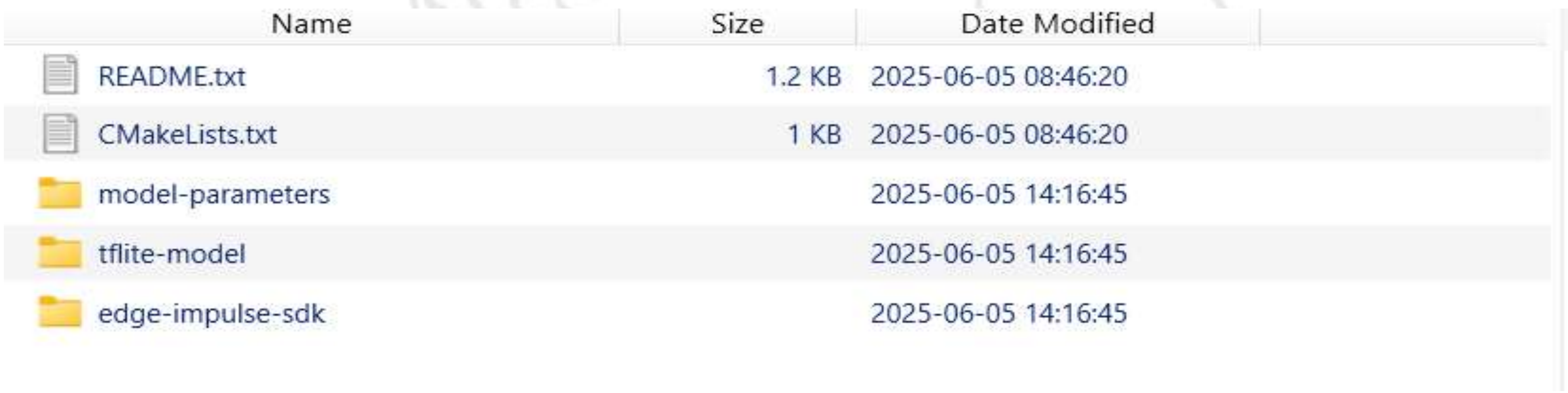
**EON™ Compiler**
Same accuracy, 17% less RAM, 22% less ROM.

**Latest build**

v3 (C++ library)
Today, 12:31:51

**Build output**

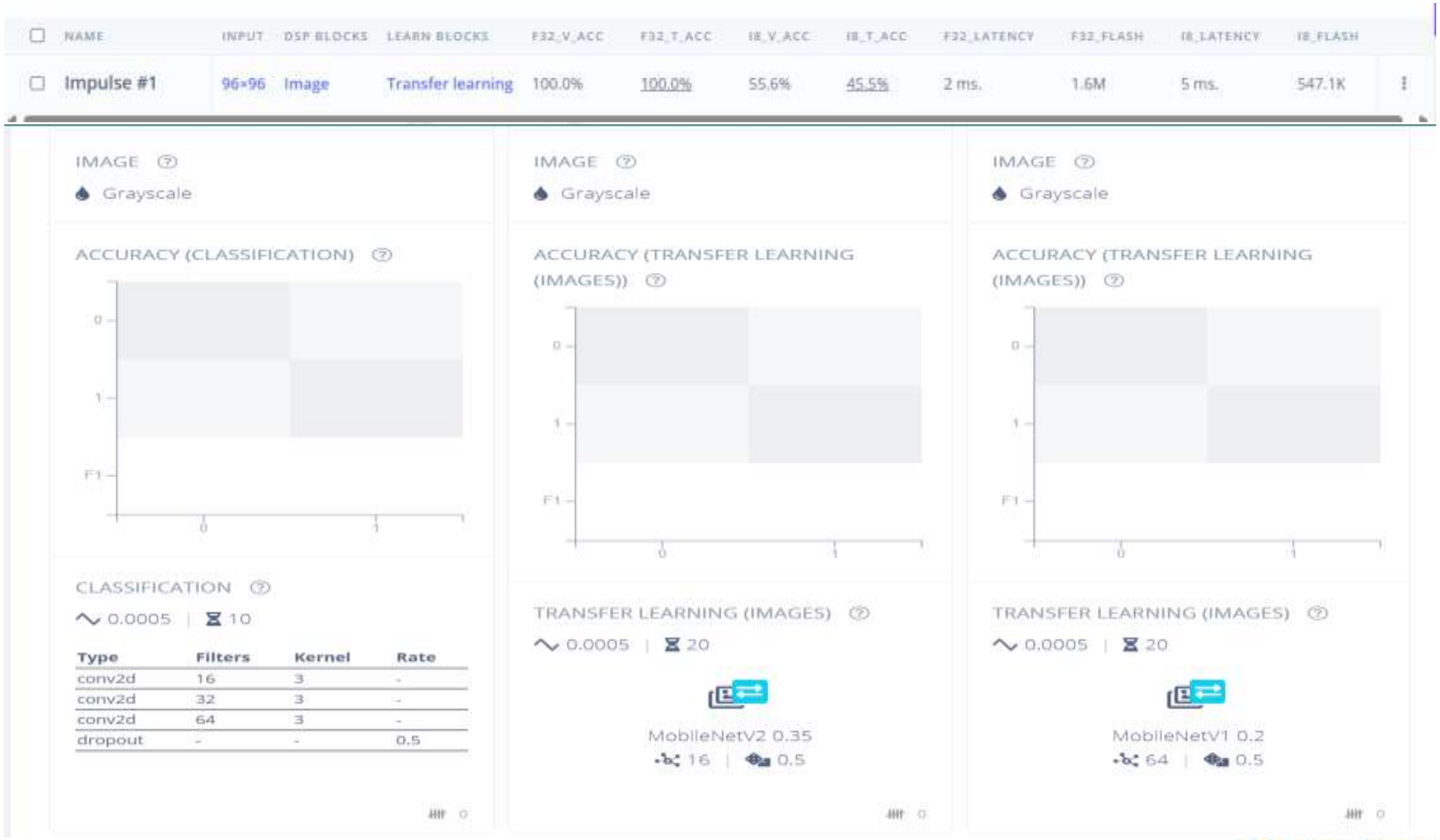
```
Creating job... OK (ID: 34062026)

✓Job scheduled at 05 Jun 2025 08:45:41
✓Job started at 05 Jun 2025 08:45:45
Writing templates...
✓Job scheduled at 05 Jun 2025 08:45:54
✓Job started at 05 Jun 2025 08:45:58
Compiling EON model...
Compiling EON model OK

Removing clutter...
Removing clutter OK

Copying output...
Copying output OK
```



| Name | Size | Date Modified |
|---|---|---|
| README.txt | 1.2 KB | 2025-06-05 08:46:20 |
| CMakeLists.txt | 1 KB | 2025-06-05 08:46:20 |
| model-parameters | | 2025-06-05 14:16:45 |
| tflite-model | | 2025-06-05 14:16:45 |
| edge-impulse-sdk | | 2025-06-05 14:16:45 |

# Link of the Model

- https://studio.edgeimpulse.com/public/713689/latest



| NAME | INPUT | DSP BLOCKS | LEARN BLOCKS | F32_V_ACC | F32_T_ACC | I8_V_ACC | I8_T_ACC | F32_LATENCY | F32_FLASH | I8_LATENCY | I8_FLASH |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Impulse #1 | 96×96 | Image | Transfer learning | 100.0% | 100.0% | 55.6% | 45.5% | 2 ms. | 1.6M | 5 ms. | 547.1K |

IMAGE
Grayscale

ACCURACY (CLASSIFICATION)

CLASSIFICATION
0.0005 | 10

| Type | Filters | Kernel | Rate |
|---|---|---|---|
| conv2d | 16 | 3 | - |
| conv2d | 32 | 3 | - |
| conv2d | 64 | 3 | - |
| dropout | - | - | 0.5 |

IMAGE
Grayscale

ACCURACY (TRANSFER LEARNING (IMAGES))

TRANSFER LEARNING (IMAGES)
0.0005 | 20
MobileNetV2 0.35
16 | 0.5

IMAGE
Grayscale

ACCURACY (TRANSFER LEARNING (IMAGES))

TRANSFER LEARNING (IMAGES)
0.0005 | 20
MobileNetV1 0.2
64 | 0.5

## Integeration with c++ library based TFLITE model(NEXT STEP)

1. **Run RISC-V Code** Without Hardware
   Whisper allows you to execute and debug RISC-V assembly or compiled C code entirely in software, simulating the behavior of the **SweRV EH1 core** without needing physical hardware or an FPGA123.
2. **Interactive Debugging and Inspection**
   You can single-step through code, inspect and modify registers, and examine simulated system memory in interactive mode, making it useful for learning and troubleshooting123.
3. **Integration with RVfpga Workflow**
   Whisper is integrated into the RVfpga toolchain, enabling you to complete labs and experiments in simulation, and can be used alongside other Verilator-based simulators for a comprehensive learning experience34.

## References

- **RVfpga HarvardX edX** and **RVfpga source code**
- TinyML foundational courses and **TensorFlow Lite Micro documentation.**
- **WorldQuant University** Applied AI Lab content
- RVfpga simulator **Vibodo,Rvfpga Nexys Board,Piplines,Whisper**(Debugging and simulation) and Tracer (**GTK wave** usage), then **whisper for c program files** compilation.
- **Report 1**,**Report 2** and **internal study materials.**