

SHRUTI PANDEY - 25MML0020

DATA ACQUISITION:

The screenshot shows the Edge Impulse Studio interface for data acquisition. The left sidebar includes options like Dashboard, Devices, Data acquisition (selected), Experiments, EON Tuner, Impulse design, Create impulse, and Image. A central panel displays a summary of data collection: "DATA COLLECTED 33 items" and "TRAIN / TEST SPL... 82% / ...". Below this is a "Dataset" section with tabs for Training (27), Test (6), and Post-processing (0). A table lists samples with columns for SAMPLE NAME, LABELS, and ADDED. The labels include "hanger" and "watch". A "Collect data" section with a "Connect a device" button and a "Click on a sample to load..." button is also present.

The screenshot shows the Edge Impulse Studio interface for impulse design. The left sidebar is identical to the previous screen. The main area is titled "Impulse #1" and contains a descriptive box: "An impulse takes raw data, uses signal processing to extract features, and then uses a learning block to classify new data." Below this are four blocks: "Image data" (red background), "Image" (white background), "Object Detection (Images)" (purple background), and "Output features" (green background). The "Image data" block has sections for Input axes (image, 96x96), Image w..., and Resize mode. The "Image" block has Name (Image) and Input axes (1). The "Object Detection" block has Name (Object detection) and Input features (Image checked). The "Output features" block lists 6 categories: hanger, keyboard, lock, monitor, mouse, and watch. A "Save Impulse" button is located at the bottom right.

EDGE IMPULSE

- Dashboard
- Devices
- Data acquisition
- Experiments
- EON Tuner
- Impulse design
 - Create impulse
 - Image
- Upgrade Plan**
Get access to higher job limits and more collaborators.

Raw data

Show: All labels Hanger.6eea93cc (hang...

Raw features

0x836b5c, 0x7d654f, 0x725843, 0x664a34, 0x593b21, 0x4d2e19, 0...

DSP result

Image

Parameters

Image

Color depth: RGB

Save parameters

EDGE IMPULSE

- Dashboard
- Devices
- Data acquisition
- Experiments
- EON Tuner
- Impulse design
 - Create impulse
 - Image
- Upgrade Plan**
Get access to higher job limits and more collaborators.

javavenom / javavenom-project-1 PERSONAL

Target: Cortex-M4F 80MHz

Neural Network settings

Training settings

Number of training cycles: 10

Use learned optimizer:

Learning rate: 0.001

Training processor: CPU

Data augmentation:

Advanced training settings

Validation set size: 20 %

Split train/validation set on metadata key:

Batch size: 32

Training output

Model Model version: Quantized (int8)

Last training performance (validation set)

F1 SCORE: 0.0%

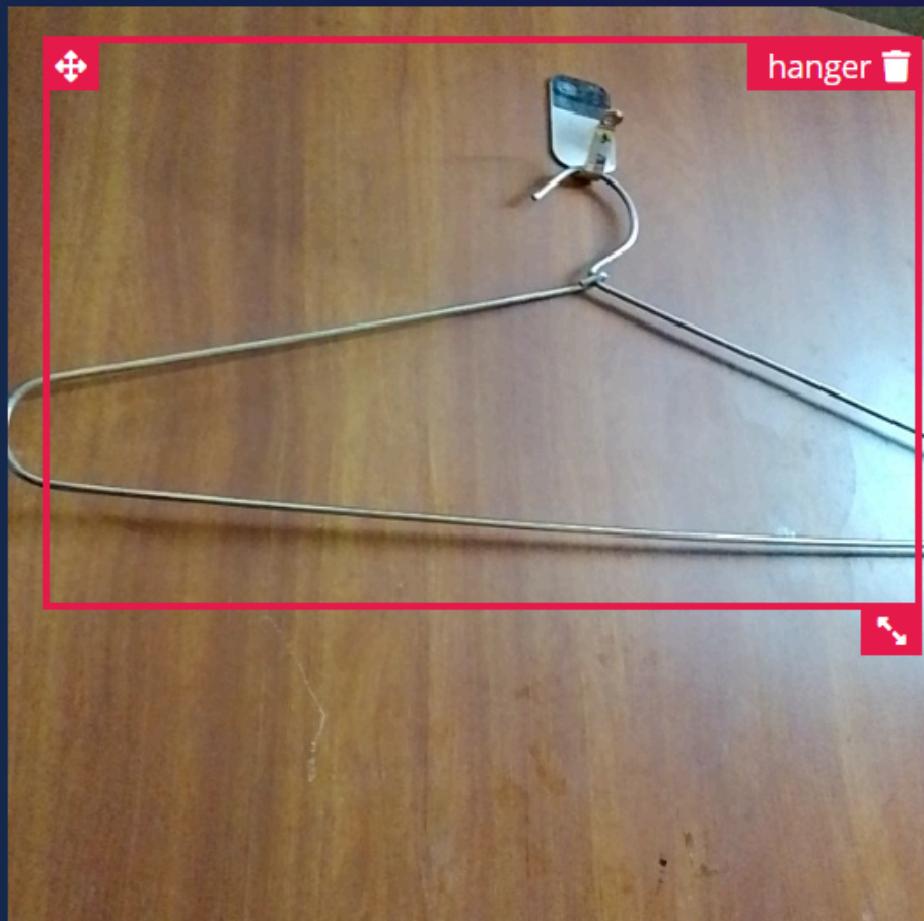
Confusion matrix (validation set)

	BACKG	HANGE	KEYBO	LOCK	MONIT	MOUSE	WATCH
BACKGRO	100%	0%	0%	0%	0%	0%	0%
HANGER	-	-	-	-	-	-	-
KEYBOAR	100%	0%	0%	0%	0%	0%	0%
LOCK	100%	0%	0%	0%	0%	0%	0%
MONITOR	100%	0%	0%	0%	0%	0%	0%
MOUSE	100%	0%	0%	0%	0%	0%	0%
WATCH	-	-	-	-	-	-	-
F1 SCORE	1.00	0.00	0.00	0.00	0.00	0.00	0.00

Sample images :

Hanger.6eea925g

⋮ ⋯

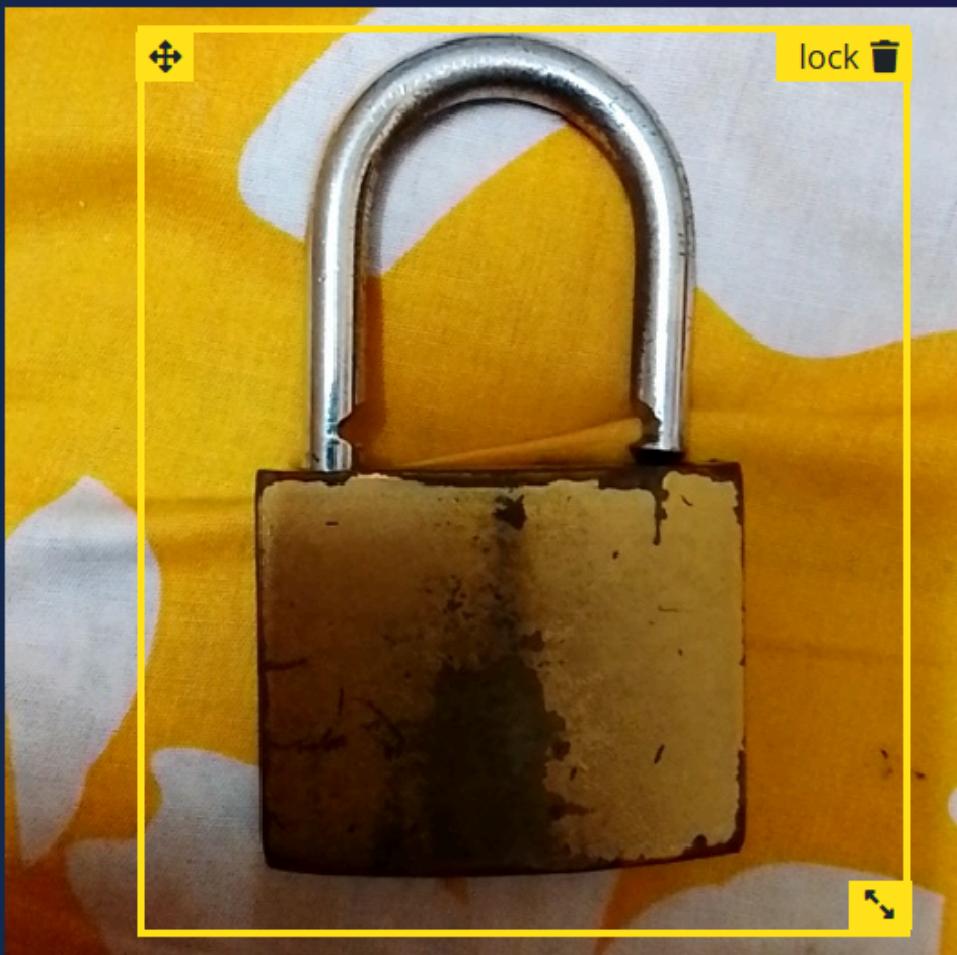


undo | rotate | zoom | search | 100% ▾

Watch.6eea4ic9

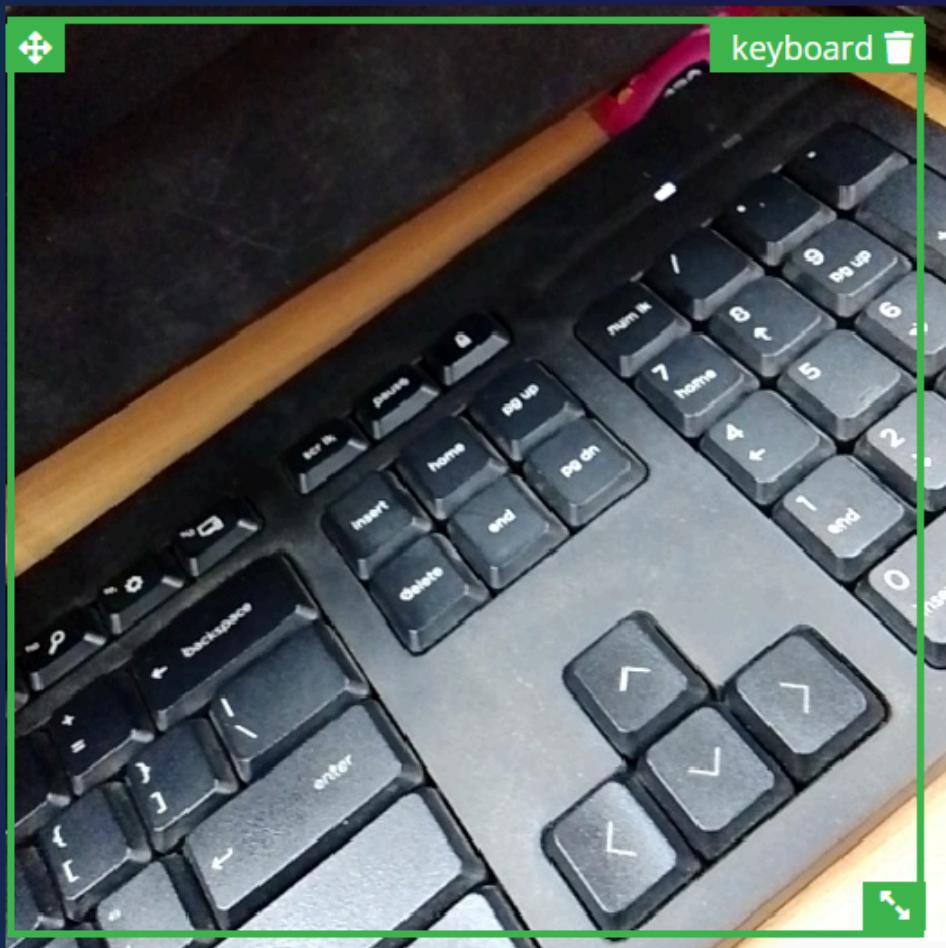


Lock.6eea09jt



100% ▾

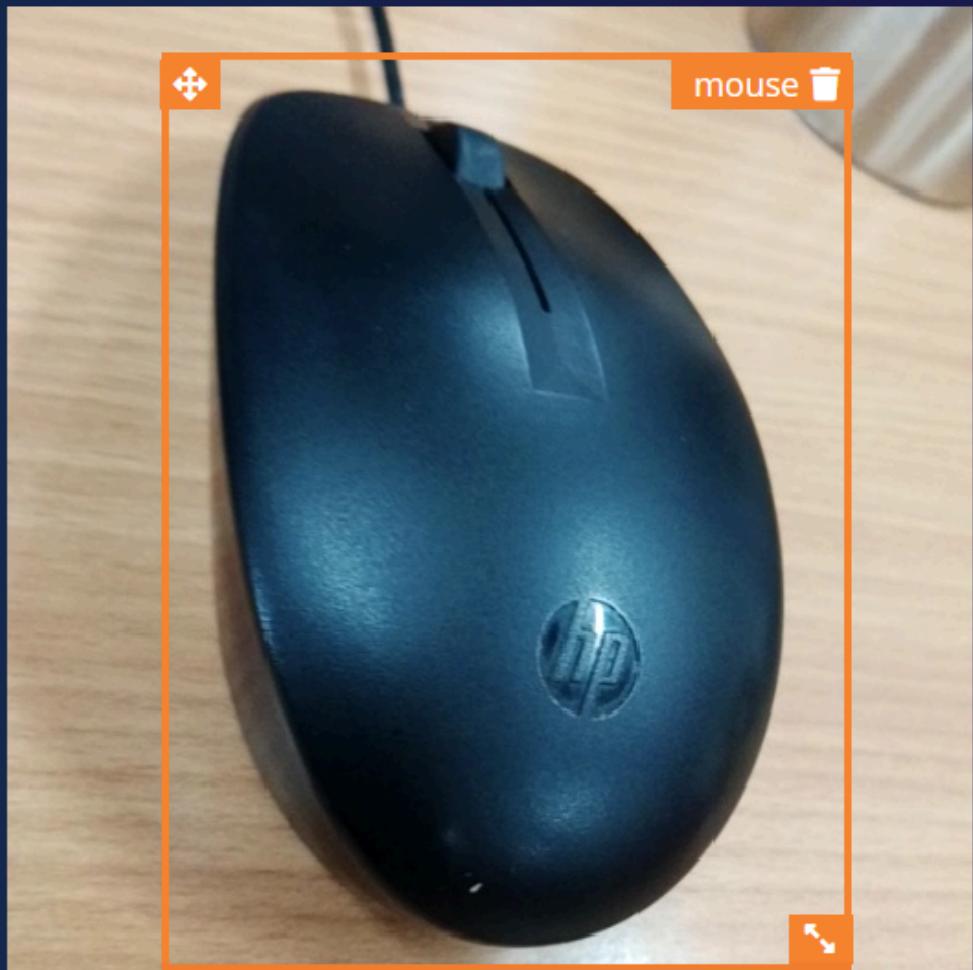
Keyboard.6ebbubdf



100% ▾

Mouse1.6ebbca7b

⋮ ⓧ



100% ▾

Monitor1.6ebb5qn3

A screenshot of a Jupyter Notebook interface titled "monitor". The notebook displays Python code for training a neural network and evaluating its performance on a test dataset.

```
history = model.fit(xtrain.reshape(4000, 32, 32), train_labels, epochs=10, verbose=1)
print("Model training complete; history stored.")

Epoch 0/10
0s/step - accuracy: 0.992 - loss: 0.000
Epoch 1/10
0s/step - accuracy: 0.998 - loss: 0.000
Epoch 2/10
0s/step - accuracy: 0.999 - loss: 0.000
Epoch 3/10
0s/step - accuracy: 0.999 - loss: 0.000
Epoch 4/10
0s/step - accuracy: 0.999 - loss: 0.000
Epoch 5/10
0s/step - accuracy: 0.999 - loss: 0.000
Epoch 6/10
0s/step - accuracy: 0.999 - loss: 0.000
Epoch 7/10
0s/step - accuracy: 0.999 - loss: 0.000
Epoch 8/10
0s/step - accuracy: 0.999 - loss: 0.000
Epoch 9/10
0s/step - accuracy: 0.999 - loss: 0.000
Epoch 10/10
0s/step - accuracy: 0.999 - loss: 0.000
Model training complete; history stored.

test_loss, test_acc = model.evaluate(test_images_reduced, test_labels, verbose=0)
print("Test loss: (test_loss: {})".format(test_loss))
print("Test accuracy: (test_acc: {})".format(test_acc))

353/353 - 0s - 3ms/step - accuracy: 0.9999 - loss: 0.0000
Test loss: 0.0000
Test accuracy: 0.9999

import matplotlib.pyplot as plt
print("matplotlib.pyplot imported as plt.")

matplotlib.pyplot imported as plt.
```

