

SHRUTI PANDEY - 25MML0020

DATA ACQUISITION:

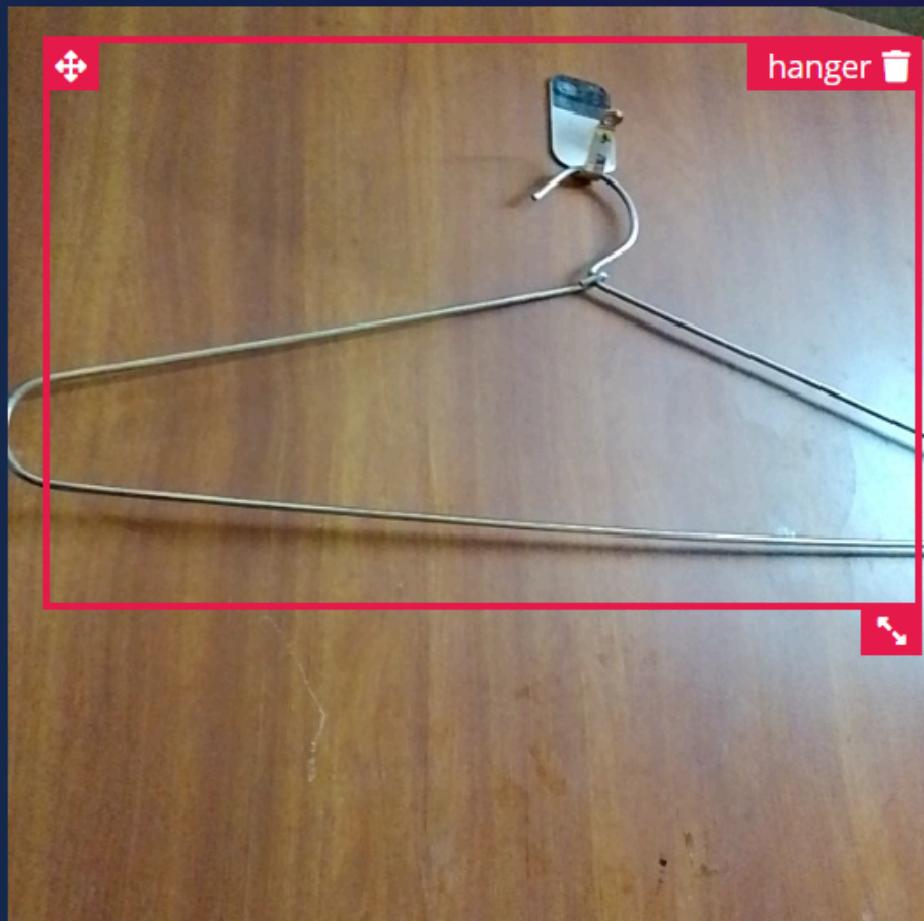
The screenshot shows the Edge Impulse Studio interface for data acquisition. The left sidebar includes options like Dashboard, Devices, Data acquisition (selected), Experiments, EON Tuner, Impulse design, Create impulse, and Image. The main area displays a dataset summary: "DATA COLLECTED 33 ite..." 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 samples listed are Hanger.6eea93cc, Hanger.6eea925g, Hanger.6eea921t, Hanger.6eea8u1m, and Watch.6eea4tdh. A prominent button at the bottom right says "Click on a sample to load...".

The screenshot shows the Edge Impulse Studio interface for impulse design. The left sidebar includes options like Dashboard, Devices, Data acquisition, Experiments, EON Tuner, Impulse design (selected), Create impulse, and Image. The main area is titled "Impulse #1" and contains a descriptive message: "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 configuration blocks: "Image data" (red background), "Image" (white background), "Object Detection (Images)" (purple background), and "Output features" (green background). The "Image data" block has settings for Input axes (image, 96x96) and Resize mode (Fit). The "Image" block has Name (Image) and Input axes (1). The "Object Detection (Images)" 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.

Sample images :

Hanger.6eea925g

⋮ ⋯

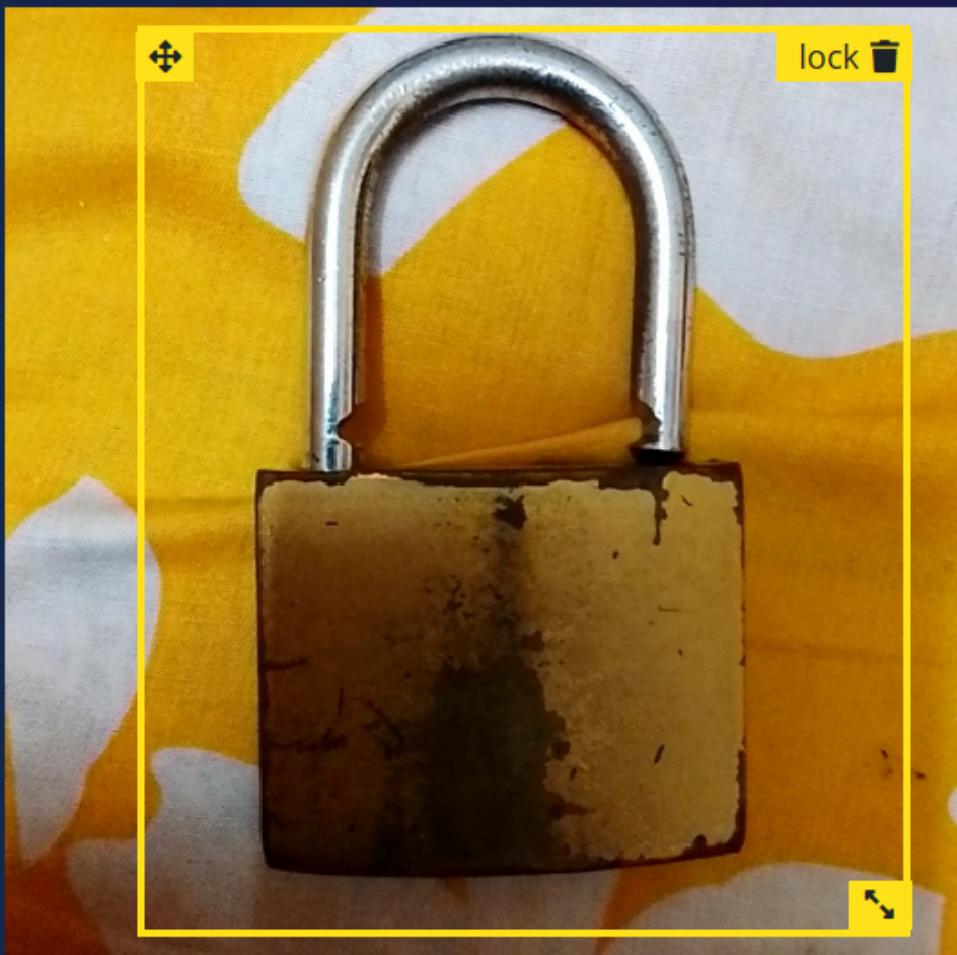


undo | rotate | zoom | search | 100% ▾

Watch.6eea4ic9

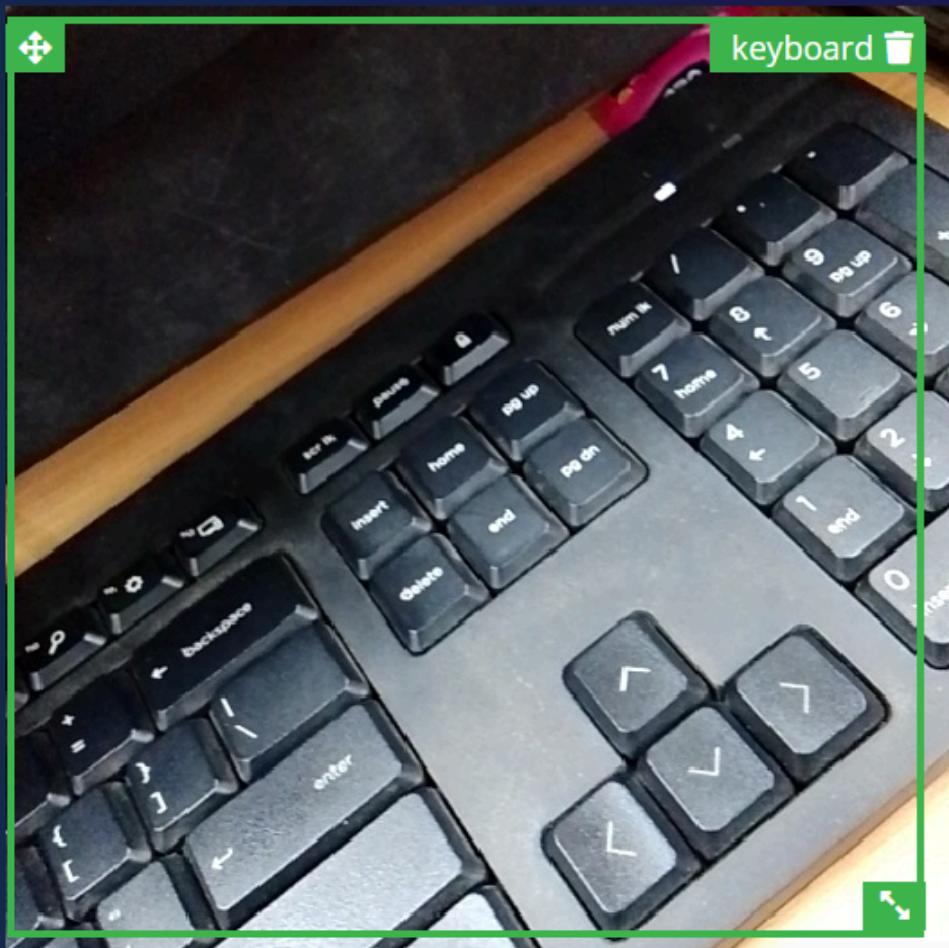


Lock.6eea09jt



100% ▾

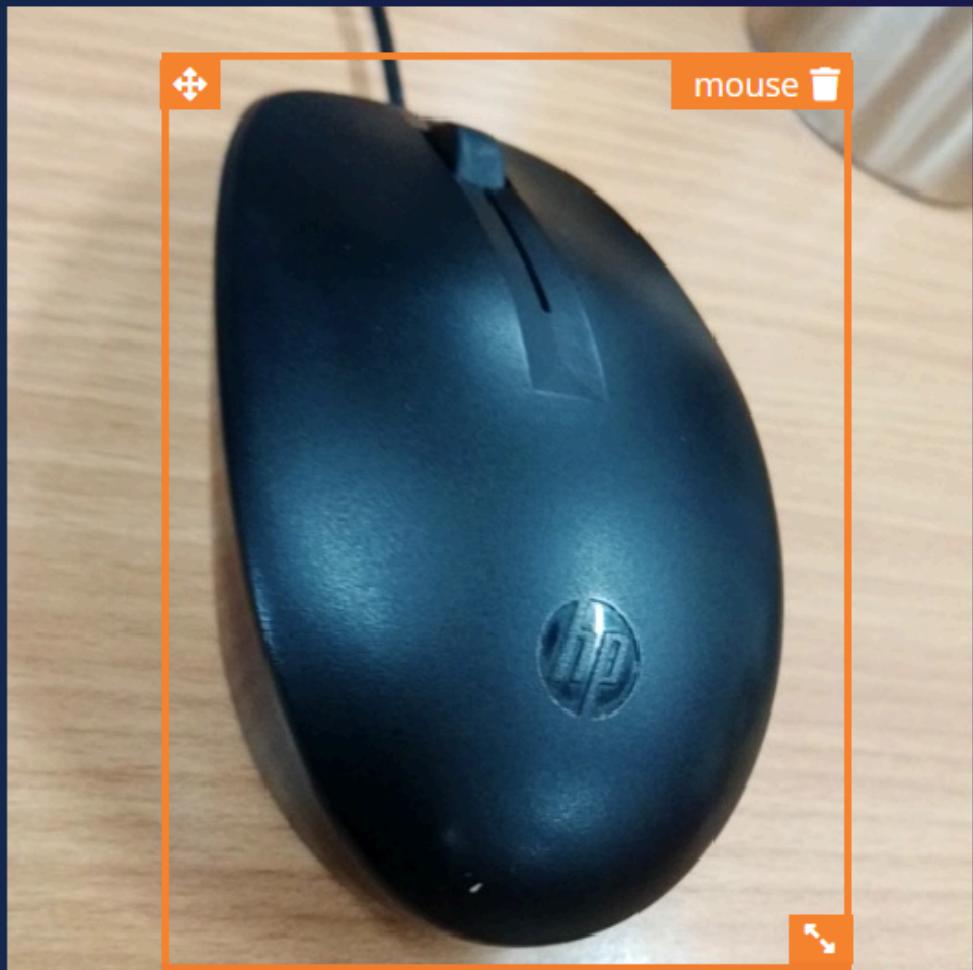
Keyboard.6ebbubdf



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.
```



