

AI Hardware, Fall 2024  
ECE 4501

# **Shaz *Animal***

*Bioacoustic Animal Identifier*



Project Group #2  
Ethan Jenkins, Bella Heintges, Eric Sheetz

# Project Motivation

*Have you ever heard an animal sound and wondered what made it?*

- Demonstrate the ***feasibility of deployed bioacoustic ID tools***
  - Automated, seamless, and accessible classification
- Leverage ***TinyML*** for efficient processing
  - Powerful machine learning in resource-constrained environments
  - Enables on-site application of audio recognition algorithms
- ***Variety of applications*** for system usage
  - Educational tool to teach children elementary concepts
  - Sustainable & practical method for ecologist wildlife monitoring

# Objectives

1. Develop a machine learning algorithm capable of processing and identifying the animal responsible for a given audio input
2. Train the model such that it is capable of classifying the sounds of **4 common animals** with at least **80% accuracy**

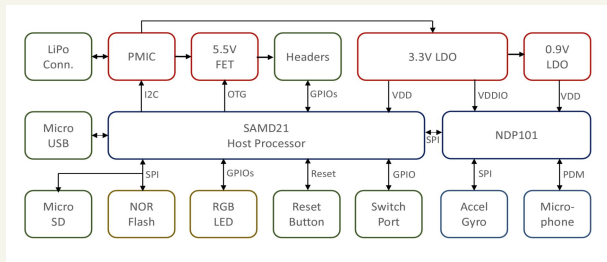


3. Integrate the program with the Syntiant TinyML board, utilizing the built-in microphone, so the system can be used on-site

# Technology Stack

## Hardware

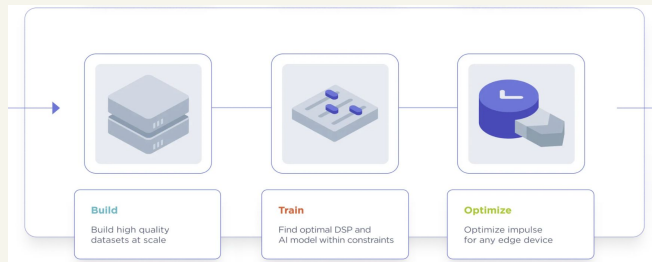
- Syntiant TinyML Board
  - NDP101
  - ARM Cortex-M4
- Linux-enabled device



*Syntiant TinyML Board block diagram*

## Software

- Code and package tools
  - git
  - edge-impulse-cli, arduino-cli
- Edge Impulse Studio



*Edge-Impulse modeling process*

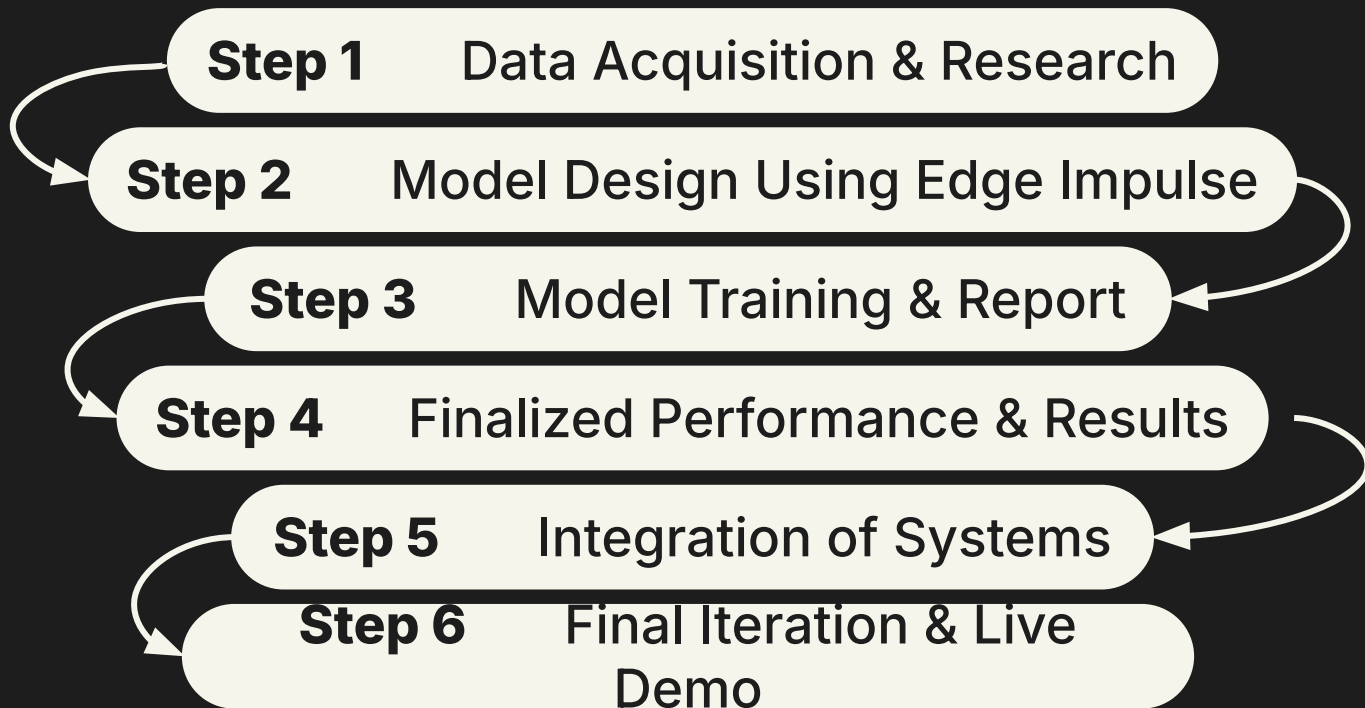
# Syntiant NDP101

## *Neural Decision Processor*

- Compute-in-memory for high parallelism and efficiency
- Memory for 589k parameters
- Low power consumption and fast inferences on 5mmx5mm chip, perfect for resource- constrained edge AI applications



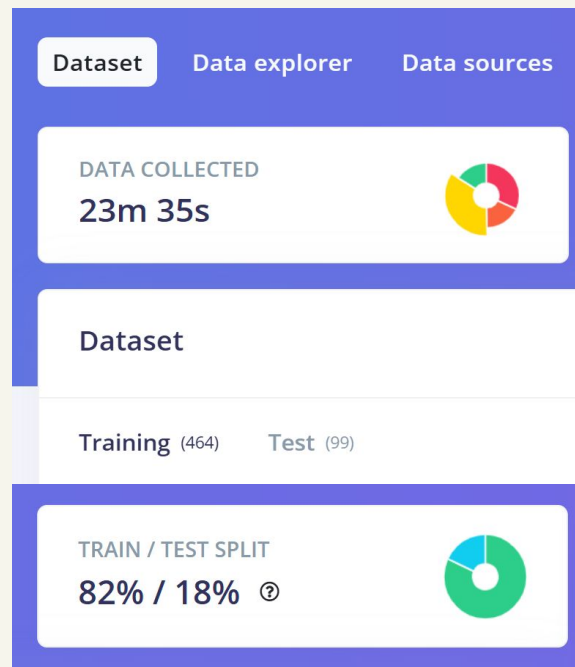
# Methodology



## Step 1

# Data Acquisition & Research

- Collected 563 total audio files from online audio libraries
  - Cat: 185 audio files
  - Cow: 84 audio files
  - Dog: 128 audio files
  - Frog: 67 audio files
- Data loaded into Edge-Impulse with an **82% / 18%** Train/Test split
  - Training: 464 audio files
  - Testing: 99 audio files




# Step 2

# Model Design


## Step 2.1


### Model Input


**Time series data**








**Input axes**  
audio

**Window size**  
  
3,000 ms.

**Window increase (stride)**  
  
1,000 ms.

**Frequency (Hz)**  
 


**Zero-pad data**  
☒




## Step 2.2


### Processing Blocks

**Audio (MFE)**




**Name**  
MFE


**Input axes (1)**  
Signal   
audio




**Spectrogram**



**Name**  
Spectrogram


**Input axes (1)**  
Signal   
audio



## Step 2.3

### Learning Block


**Classification**



**Name**

**Input features**  
☒ MFE  
☒ Spectrogram

**Output features**  
4 (Cat, Cow, Dog, Frog)





## Step 2.1

# Definition of Model Input

- Audio files varied in length, generally ranging between 2 to 5 seconds
- Analyzed a 3 second (3000 ms) window of each file
  - Window stride of 1 second (1000ms), for files longer than the window
- Nominal sampling frequency common for digital audio

The image shows a configuration interface for 'Time series data' on a red background. It includes several settings:

- Input axes:** Set to 'audio'.
- Window size:** A slider set to 3,000 ms.
- Window increase (stride):** A slider set to 1,000 ms.
- Frequency (Hz):** A text input field containing '44100' with a refresh icon.
- Zero-pad data:** A checkbox that is checked.

Each setting has a help icon (question mark) to its right.

## Step 2.2

# Design of Processing Blocks

*Used 2 processing blocks to analyze the audio data:*

### **Block 1: Mel Frequency Energy (MFE)**

- Extracts audio signal features
- Emphasizes frequencies important for audio classification

### **Block 2: Spectrogram**

- Visualizes the audio frequencies over time
- Detailed representation for training and testing

**Parameters** [Autotune parameters](#)

Mel-filterbank energy features

Frame length ⓘ	<input type="text" value="0.02"/>
Frame stride ⓘ	<input type="text" value="0.01"/>
Filter number ⓘ	<input type="text" value="32"/>
FFT length ⓘ	<input type="text" value="512"/>
Low frequency ⓘ	<input type="text" value="0"/>
High frequency ⓘ	<input type="text" value="22050"/>

**Parameters** [Autotune parameters](#)

Spectrogram

Frame length ⓘ	<input type="text" value="0.02"/>
Frame stride ⓘ	<input type="text" value="0.01"/>
FFT length ⓘ	<input type="text" value="256"/>

Normalization

Noise floor (dB) ⓘ	<input type="text" value="-60"/>
--------------------	----------------------------------

## Step 2.3

# Design of Learning Block

- Chose a **Classification** learning block
  - Trains the model to distinguish and classify the different sounds

Neural Network settings

Training settings

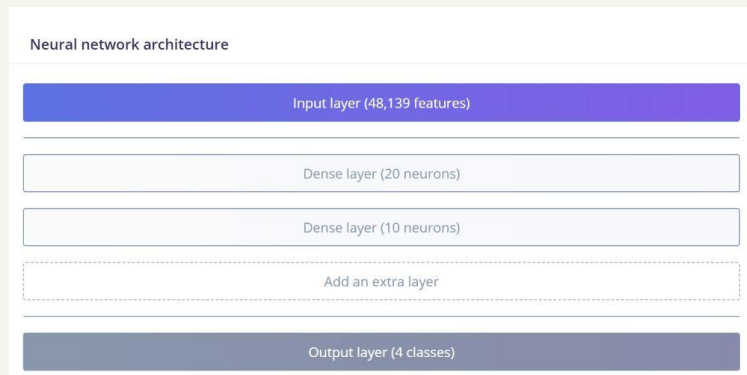
Number of training cycles ⓘ

Use learned optimizer ⓘ ☐

Learning rate ⓘ

Training processor ⓘ

***Input:** Processed features (48,139) from the MFE and Spectrogram*



***Output:** Classification of each of the audio files (563) into one of the 4 animals*

## Step 3

# Model Training

- Conducted 3 training iterations
- Model trained for 50 epochs
  - Saved best model results for later usage
- Reported training **accuracy of 86.4%** and a **loss of 0.48**
  - Surpasses our targeted model training accuracy of 80%
  - Ready for hardware integration

### Training output

```
p
Epoch 47/50
15/15 - 0s - loss: 0.3563 - accuracy: 0.8792 - val_loss: 0.4802 - val_accuracy: 0.8644 - 352ms/epoch - 23ms/step
p
Epoch 48/50
15/15 - 0s - loss: 0.3606 - accuracy: 0.8750 - val_loss: 0.6769 - val_accuracy: 0.7288 - 377ms/epoch - 25ms/step
p
Epoch 49/50
15/15 - 0s - loss: 0.3760 - accuracy: 0.8369 - val_loss: 0.5295 - val_accuracy: 0.7966 - 399ms/epoch - 27ms/step
p
Epoch 50/50
15/15 - 0s - loss: 0.3266 - accuracy: 0.8771 - val_loss: 0.5547 - val_accuracy: 0.8220 - 394ms/epoch - 26ms/step
p
Finished training

Saving best performing model... (based on validation loss)
```

### Model

Model version: ⑦ Quantized (int8) ▾

#### Last training performance (validation set)

 ACCURACY  
**86.4%**

 LOSS  
**0.48**

#### Confusion matrix (validation set)

	CAT	COW	DOG	FROG
CAT	87.9%	3.0%	6.1%	3.0%
COW	4.8%	76.2%	19.0%	0%
DOG	0%	6.8%	93.2%	0%
FROG	0%	5%	15%	80%
F1 SCORE	0.92	0.76	0.87	0.86

## Step 4

# Model Performance & Results



*Accuracy = 86.4%, Loss = 0.48*

- Accurate classifications for the majority of data
- Some errors depicted within class clusters
  - Limited training data
  - Lack of preprocessing

On-device performance [?](#)

Engine: [?](#) **EON™ Compiler** [▼](#)



INFERENCE TIME  
19 ms.



PEAK RAM USAGE  
48.4K



FLASH USAGE  
955.5K

## Step 5

# Integration with Hardware

### *Current status:*

- Now necessary to **flash** the system to the Syntiant board
- Experiencing difficulties in establishing communication

```
xvj5pd@DESKTOP-LON0SQL:~/firmware-syntiant-tinyml$ ./arduino-build.sh --flash
Error during Upload: no Fully Qualified Board Name provided
xvj5pd@DESKTOP-LON0SQL:~/firmware-syntiant-tinyml$ ls
LICENSE                               firmware-syntiant-tinyml.ino.bin
LICENSE-apache-2.0.txt                firmware-syntiant-tinyml.ino.elf
README.md                             firmware-syntiant-tinyml.ino.hex
arduino-build.sh                      firmware-syntiant-tinyml.ino.map
arduino-win-build.bat                 firmware-syntiant-tinyml.ino.with_bootloader.bin
firmware-syntiant-tinyml.ino          firmware-syntiant-tinyml.ino.with_bootloader.hex
xvj5pd@DESKTOP-LON0SQL:~/firmware-syntiant-tinyml$ arduino-cli board list
No boards found.
xvj5pd@DESKTOP-LON0SQL:~/firmware-syntiant-tinyml$
```

*Connection with board error*

### *Next steps:*

- Ensure relevant tool files are accessible in Ubuntu
  - Reconfigure Syntiant link
- Explore device workarounds

```
root@DESKTOP-27I3R5C:/mnt/C/Users/Eric/Downloads/clone_syntiant-rc-go-stop-syntiant-ndp101-v41# ./flash_linux.sh

A new release of Arduino CLI is available: 0.19.0 - 1.1.1
https://arduino.github.io/arduino-cli/latest/installation/#latest-packages

A new release of Arduino CLI is available: 0.19.0 - 1.1.1
https://arduino.github.io/arduino-cli/latest/installation/#latest-packages

A new release of Arduino CLI is available: 0.19.0 - 1.1.1
https://arduino.github.io/arduino-cli/latest/installation/#latest-packages
You're using an untested version of Arduino CLI, this might cause issues (found: 0.19.0, expected: 0.13.x)
Finding Arduino SAMD core v1.8.9...
Finding Arduino SAMD OK
Finding Arduino MKRZero...
Cannot find a connected Arduino MKRZero development board (via 'arduino-cli board list').
If your board is connected, double-tap on the RESET button to bring the board in recovery mode.
root@DESKTOP-27I3R5C:/mnt/C/Users/Eric/Downloads/clone_syntiant-rc-go-stop-syntiant-ndp101-v41#
```

*Project flash error*

## Step 6

# Final Iteration & Live Demo

- Once hardware connection is established, the serial output will be used for real-time classification results
- Digital playback will be used so sufficient trials can be manually executed to calculate accuracy of on-board model
- Expect similar (slightly worse?) performance compared to test results, as speaker/microphone noise may reduce audio clarity

# Final Thoughts

## *Lessons learned:*

- Processing blocks greatly affect the accuracy of the model
  - Spectrogram processing alone does not accurately classify audio
- Necessary to implement a robust dataset for both testing and training processes

## *Conclusions:*

- System successfully designed and capable of classifying animal calls
  - Classes defined and tested
  - Accuracy greater than 80% with minimal loss
- On track to successful integration using the Syntiant TinyML Hardware



# Expansion & Improvements

- Possible to increase the accuracy of the model by acquiring additional audio files per animal (*e.g. 500/animal*)
  - Larger dataset, more robust training capabilities
- Add additional classifiers to expand beyond current scope
  - Requires a search for additional audio files
  - More realistic on-site system implementation
- Include another processing block to streamline identification
- Integrate a visual aid component for better user accessibility



***Thank you!***