

1. Questions

Q1: Does the model perform as accurately as expected on your smartphone? List a few methods to improve the model's accuracy.

Answer:

- **Model Performance on Smartphone:**
Yes, the model performed accurately on the smartphone. The model detected keywords Hello World reliably under normal conditions, but struggled in noisy environments.
 - **Methods to Improve Accuracy:**
 1. **Increase Dataset Diversity:** Use a more diverse dataset, including variations in accents, background noise, and speaker demographics.
 2. **Data Augmentation:** Introduce techniques like pitch shifting, adding synthetic noise, and time-stretching to enhance robustness.
 3. **Hyperparameter Tuning:** Optimize parameters like the learning rate, batch size, and the number of epochs during training.
 4. **Model Architecture Improvement:** Experiment with advanced architectures or pre-trained models suitable for TinyML.
-

Q2: When building a model for resource-limited hardware, how do you balance fast inference times with acceptable model accuracy? What trade-offs did you encounter?

Answer:

- **Balancing Inference Time and Accuracy:**
To balance inference speed with accuracy, I opted for INT8 quantization, which reduces the model's size and computational demands while slightly sacrificing precision.
 - **Trade-offs Encountered:**
 1. **Reduced Accuracy:** Quantization and smaller model sizes led to a slight decrease in accuracy, especially for harder-to-detect keywords.
 2. **Increased Latency in Debugging:** Optimized models require careful tuning to ensure functionality without loss of key features.
 3. **Hardware Constraints:** The limited computational power and memory of the Arduino Nano 33 BLE Sense meant I had to simplify the model architecture.
-

2. Screenshots of Training Performance

- **Screenshot 1: Training data summary**

hplp/intr x Mobile p x Respondi x KRATOS2 x ChatGPT x +

https://studio.edgeimpulse.com/studio/607743/acquisition/training?page=1

Rishabh Kulshrestha / KRATOS21-project-1 PERSONAL

Dataset

Data explorer

Data sources

AI labeling

NEW

CSV Wizard

Dataset

Training (1,166) Test (280)

ID	S. LABEL	ADDED	LENGTH	DEVICE	SIG...	CREATED	SENSORS
1582974299	u. unkno...	Yester...	1s		HS...	Yester...	audio @ ...
1582974298	u. unkno...	Yester...	1s		HS...	Yester...	audio @ ...
1582974297	u. unkno...	Yester...	1s		HS...	Yester...	audio @ ...
1582974296	u. unkno...	Yester...	1s		HS...	Yester...	audio @ ...
1582974294	u. unkno...	Yester...	1s		HS...	Yester...	audio @ ...
1582974293	u. unkno...	Yester...	1s		HS...	Yester...	audio @ ...
1582974292	u. unkno...	Yester...	1s		HS...	Yester...	audio @ ...
1582974291	u. unkno...	Yester...	1s		HS...	Yester...	audio @ ...
1582974289	u. unkno...	Yester...	1s		HS...	Yester...	audio @ ...
1582974288	u. unkno...	Yester...	1s		HS...	Yester...	audio @ ...
1582974287	u. unkno...	Yester...	1s		HS...	Yester...	audio @ ...
1582974286	u. unkno...	Yester...	1s		HS...	Yester...	audio @ ...

<

1

2

3

4

5

6

...

98

>

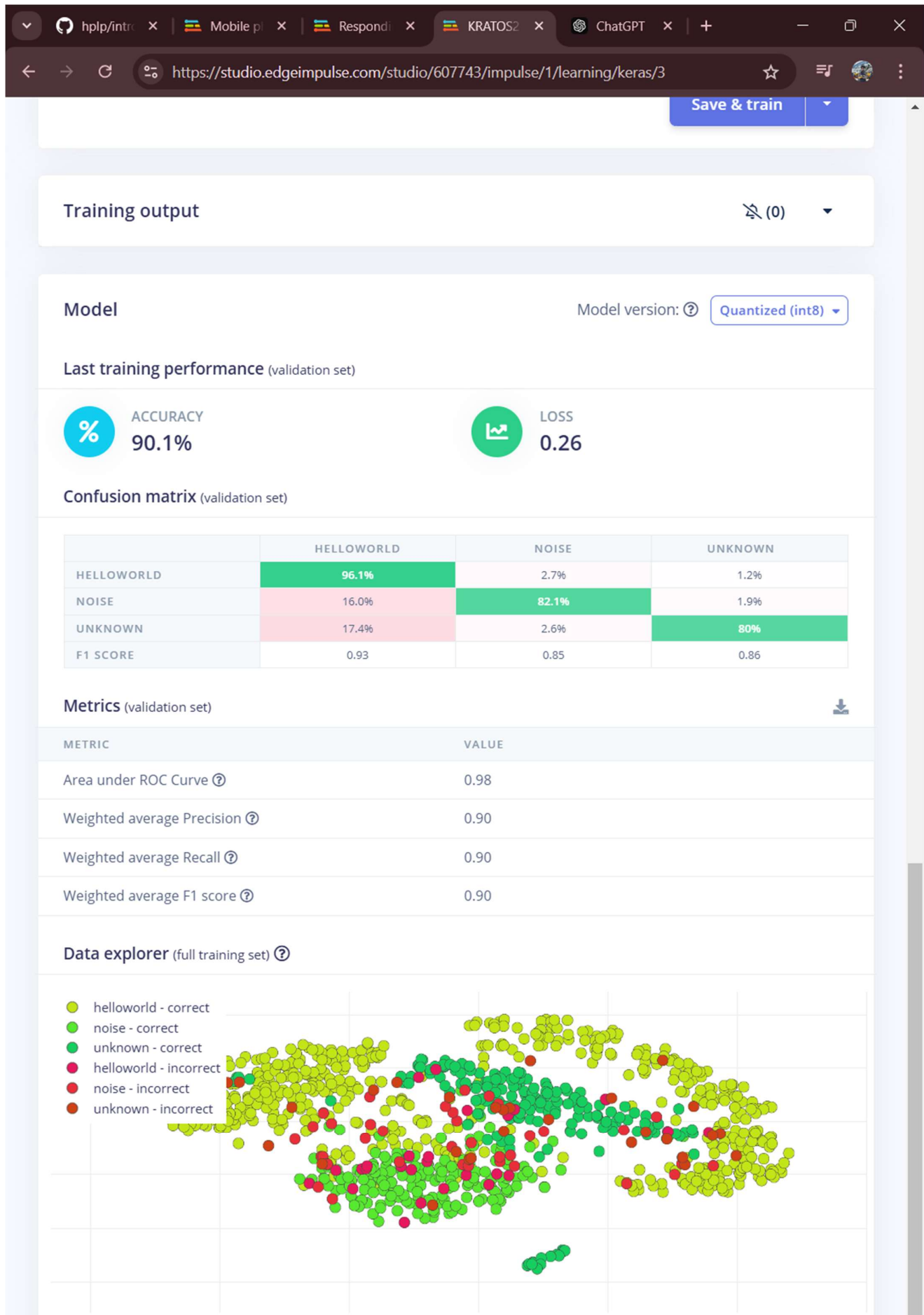
DATA COLLECTED

44m 21s

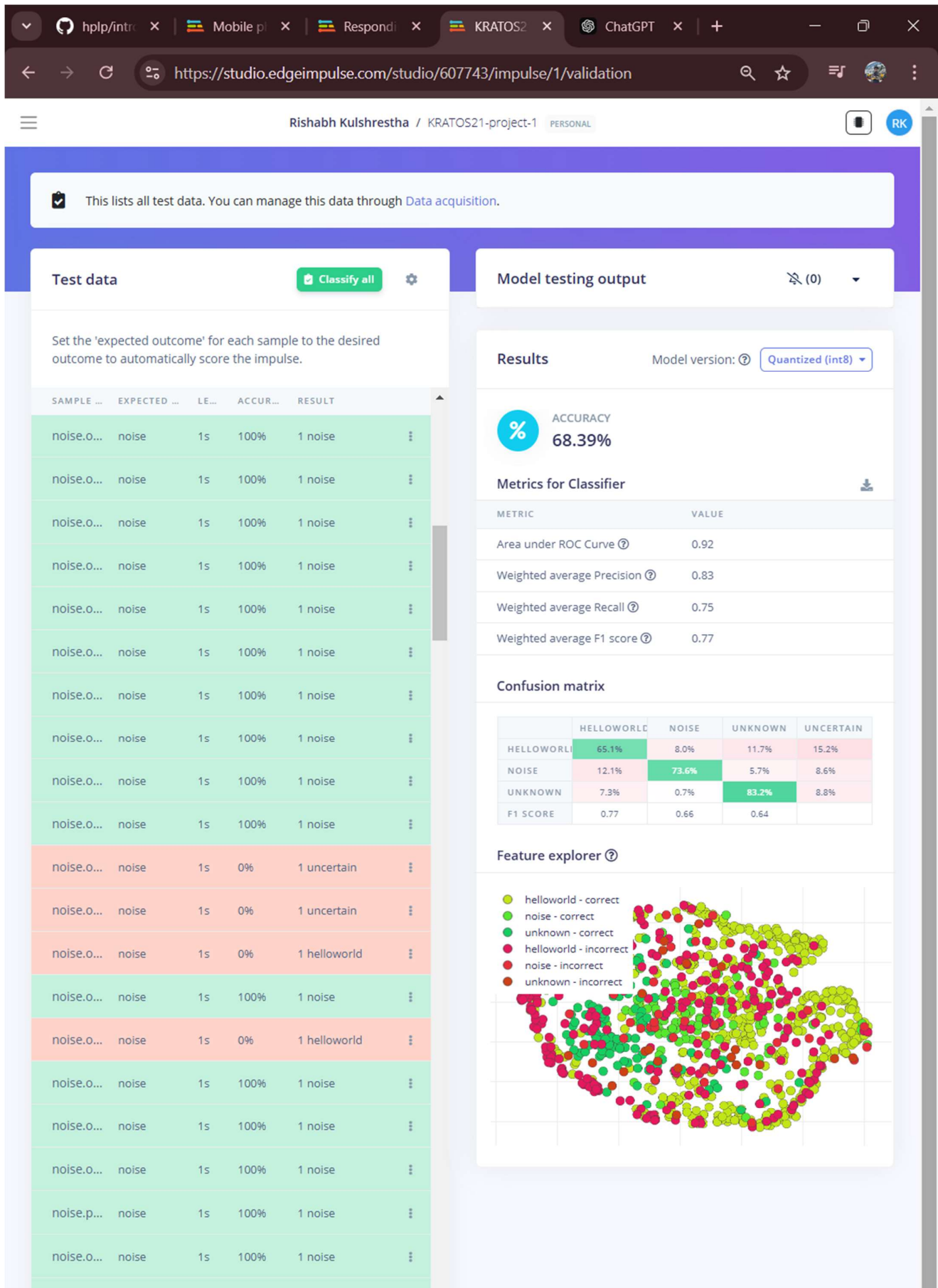
TRAIN / TEST SPLIT

74% / 26%

- **Screenshot 2:** Model accuracy and loss curves after training completion.



• **Screenshot 3: Model evaluation performance.**



- **Screenshot 4:** Edge Impulse Deployment page showing the selection of INT8 quantization.

hplp/intrc x Mobile p x Respondi x KRATOS2 x ChatGPT x +

https://studio.edgeimpulse.com/studio/607743/impulse/1/deployment

☆ 🎵 🌐 ⋮

☰

Rishabh Kulshrestha / KRATOS21-project-1 PERSONAL

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
Arduino library
An Arduino library with examples that runs on most Arm-based Arduino development boards.

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

EON™ Compiler
Same accuracy, 40% less RAM, 49% less ROM.

▼

Quantized (int8)
Selected ▼

	MFCC	CLASSIFIER	TOTAL
LATENCY	258 ms.	4 ms.	262 ms.
RAM	15.4K	3.8K	15.4K
FLASH	-	31.9K	-
ACCURACY			68.39%

Unoptimized (float32)
Select

	MFCC	CLASSIFIER	TOTAL
LATENCY	258 ms.	117 ms.	375 ms.
RAM	15.4K	7.0K	15.4K
FLASH	-	28.1K	-
ACCURACY			68.12%

Estimate for Arduino Nano 33 BLE Sense (Cortex-M4F 64MHz) - [Change target](#)

Build

3. Videos

Links to Videos:

1. **Keyword-Spotting Model on Smartphone:**

Screenshot of mobile device connected to the edgeplus model and video showing the model running on the smartphone and correctly identifying spoken keywords.

Link: [Keyword-Spotting Model on Smartphone](#)

<https://drive.google.com/drive/folders/18rZ7eRGpJRK9KCaFSCYXKz5Gzo9ya-Bi?usp=sharing>

2. **Keyword-Spotting Model on Arduino Nano 33 BLE Sense:**

Video demonstrating the model deployed on the Arduino board, showing the serial monitor output or LED control via voice commands.

Link: [Keyword-Spotting Model on Arduino Nano 33 BLE Sense](#)

<https://drive.google.com/drive/folders/18rZ7eRGpJRK9KCaFSCYXKz5Gzo9ya-Bi?usp=sharing>

4. Reflections

- **Experience Deploying to Smartphone:**

"The deployment to the smartphone was straightforward, as Edge Impulse provides clear instructions. The mobile microphone captured high-quality inputs, which contributed to better keyword detection."

- **Experience Deploying to Arduino:**

"Deploying the model to the Arduino Nano 33 BLE Sense was a rewarding challenge. First-time compilation took longer than expected (~20 minutes), and I faced difficulties loading the ZIP library into Arduino IDE initially. Adjusting the model's memory allocation and debugging issues related to the serial monitor were the most time-consuming tasks."

- **Technical Difficulties and Observations:**

1. The need to re-enter the Arduino IDE after adding the library was a surprising step.
2. The model's response time on Arduino was slightly slower than on the smartphone due to hardware limitations.
3. Noisy environments significantly impacted the detection quality on both platforms.