

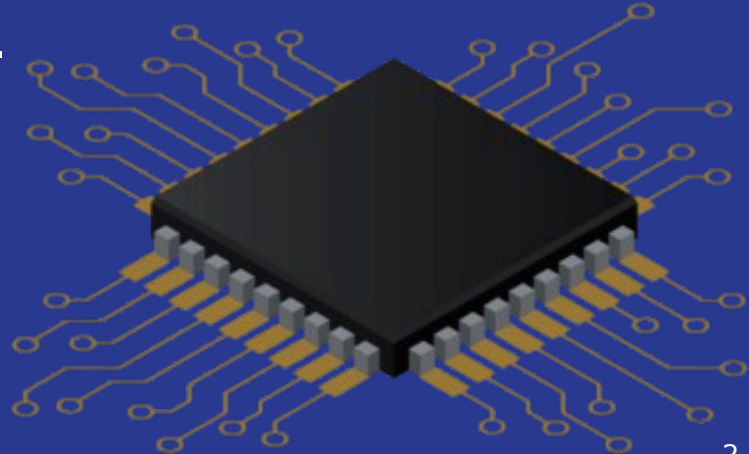
# Estación de Monitoreo Inteligente Multisensorial de Seguridad Basado en Machine Learning



Marco Vásquez Ovarés  
Junior Ruiz Sánchez

# Estación de Monitoreo Inteligente Multisensorial de Seguridad Basado en Machine Learning

Sistema embebido para detectar sonidos críticos  
usando ML.  
Seguridad más accesible e inteligente.



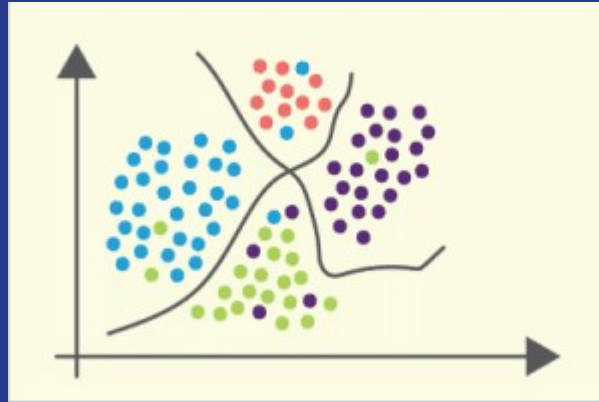
# Motivación

La seguridad es una prioridad.  
Sistemas tradicionales no detectan bien el contexto sonoro.



# Objetivo del sistema

Clasificar sonidos del entorno y activar respuestas locales



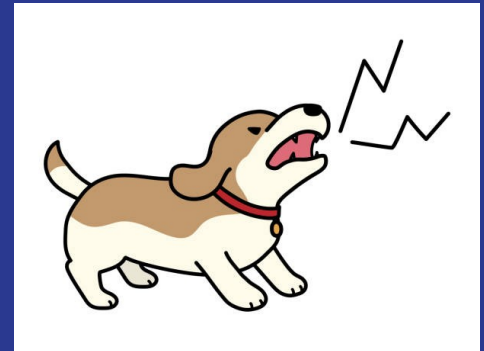
# ¿Qué sonidos detecta?



- ambiente normal
- sirenas de policía



- vidrios rotos
- ladridos de perro



# ¿Por qué esos sonidos?

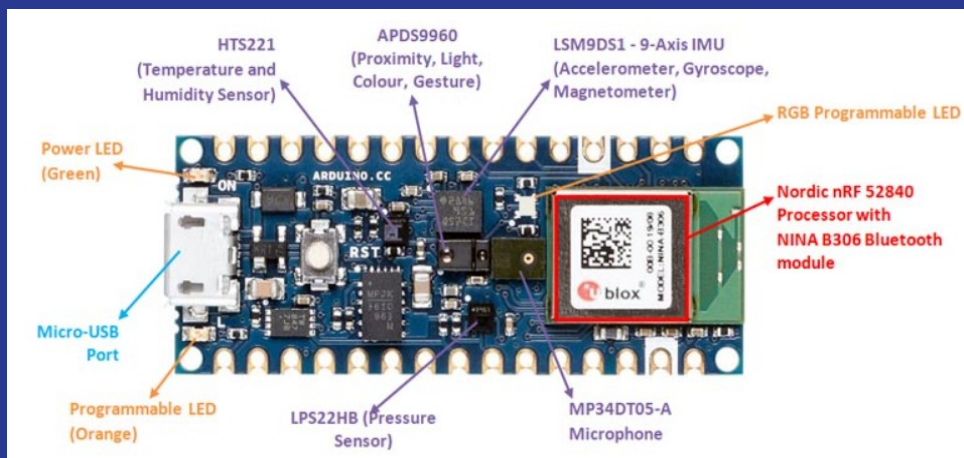
Son frecuentes en eventos de seguridad y fáciles de identificar acústicamente.



# Hardware

## Arduino Nano 33 BLE Sense.

- micrófono MEMS para capturar audio.



# Modelo de clasificación

Se entrenó un modelo en Edge Impulse para reconocer los 4 sonidos definidos.

DATA COLLECTED

13m 20s



TRAIN / TEST SPLIT

81% / 19% ?





# Recolección de datos

Se grabaron muestras con el micrófono del Arduino y se procesaron en Python.

```
Presiona ENTER para grabar muestra 2/50 de 'ladrido'...
-> Grabando...
64000 muestras capturadas.
Muestra 2 guardada: ladrido_02.wav

Presiona ENTER para grabar muestra 3/50 de 'ladrido'...
-> Grabando...
64000 muestras capturadas.
Muestra 3 guardada: ladrido_03.wav

Presiona ENTER para grabar muestra 4/50 de 'ladrido'...
-> Grabando...
64000 muestras capturadas.
Muestra 4 guardada: ladrido_04.wav

Presiona ENTER para grabar muestra 5/50 de 'ladrido'...
-> Grabando...
64000 muestras capturadas.
Muestra 5 guardada: ladrido_05.wav

Presiona ENTER para grabar muestra 6/50 de 'ladrido'...
-> Grabando...
64000 muestras capturadas.
Muestra 6 guardada: ladrido_06.wav

Presiona ENTER para grabar muestra 7/50 de 'ladrido'...
-> Grabando...
64000 muestras capturadas.
Muestra 7 guardada: ladrido_07.wav

Presiona ENTER para grabar muestra 8/50 de 'ladrido'...
-> Grabando...
64000 muestras capturadas.
Muestra 8 guardada: ladrido_08.wav

Presiona ENTER para grabar muestra 9/50 de 'ladrido'...
-> Grabando...
64000 muestras capturadas.
Muestra 9 guardada: ladrido_09.wav
```

```
-> Grabando...
64000 muestras capturadas.
Muestra 46 guardada: vidrios_46.wav

Presiona ENTER para grabar muestra 47/50 de 'vidrios'...
-> Grabando...
64000 muestras capturadas.
Muestra 47 guardada: vidrios_47.wav

Presiona ENTER para grabar muestra 48/50 de 'vidrios'...
-> Grabando...
64000 muestras capturadas.
Muestra 48 guardada: vidrios_48.wav

Presiona ENTER para grabar muestra 49/50 de 'vidrios'...
-> Grabando...
64000 muestras capturadas.
Muestra 49 guardada: vidrios_49.wav

Presiona ENTER para grabar muestra 50/50 de 'vidrios'...
-> Grabando...
64000 muestras capturadas.
Muestra 50 guardada: vidrios_50.wav

Grabacion finalizada.
+ audio [main] * python3 data.audio.py
Nombre de la clase (ej: ladrido): sirena_police
¿Cuántas muestras a grabar?: 50
Conectado a /dev/ttyACM0 a 230400 baudios.

Presiona ENTER para grabar muestra 1/50 de 'sirena_police'...
-> Grabando...
64000 muestras capturadas.
Muestra 1 guardada: sirena_police_01.wav

Presiona ENTER para grabar muestra 2/50 de 'sirena_police'...
-> Grabando...
64000 muestras capturadas.
Muestra 2 guardada: sirena_police_02.wav

Presiona ENTER para grabar muestra 3/50 de 'sirena_police'...
```

# Formato de datos

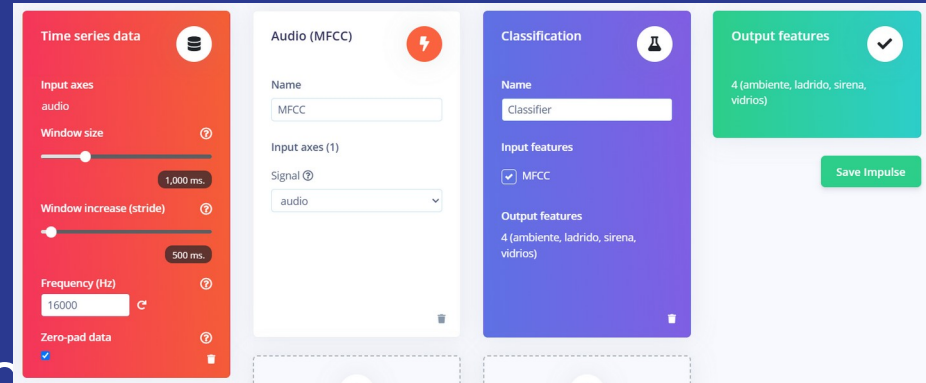
Audios .WAV de 4 segundos a 16 kHz. Organizados por clases para el entrenamiento.

SAMPLE NAME	LABEL	ADDED	LENGTH	
vidrios_02	vidrios	Yesterday, 15:...	4s	⋮
vidrios_08	vidrios	Yesterday, 15:...	4s	⋮
vidrios_05	vidrios	Yesterday, 15:...	4s	⋮
vidrios_01	vidrios	Yesterday, 15:...	4s	⋮
sirena_polic_45	sirena	Yesterday, 15:...	4s	⋮
sirena_polic_48	sirena	Yesterday, 15:...	4s	⋮
sirena_polic_50	sirena	Yesterday, 15:...	4s	⋮
sirena_polic_49	sirena	Yesterday, 15:...	4s	⋮
sirena_polic_42	sirena	Yesterday, 15:...	4s	⋮
sirena_polic_33	sirena	Yesterday, 15:...	4s	⋮
sirena_polic_36	sirena	Yesterday, 15:...	4s	⋮
sirena_polic_35	sirena	Yesterday, 15:...	4s	⋮

SAMPLE NAME	LABEL	ADDED	LENGTH	
ladrido_07	ladrido	Yesterday, 15:...	4s	⋮
ladrido_02	ladrido	Yesterday, 15:...	4s	⋮
ambiente_46	ambiente	Yesterday, 15:...	4s	⋮
ambiente_43	ambiente	Yesterday, 15:...	4s	⋮
ambiente_48	ambiente	Yesterday, 15:...	4s	⋮
ambiente_45	ambiente	Yesterday, 15:...	4s	⋮
ambiente_50	ambiente	Yesterday, 15:...	4s	⋮
ambiente_40	ambiente	Yesterday, 15:...	4s	⋮
ambiente_49	ambiente	Yesterday, 15:...	4s	⋮
ambiente_47	ambiente	Yesterday, 15:...	4s	⋮
ambiente_34	ambiente	Yesterday, 15:...	4s	⋮
ambiente_31	ambiente	Yesterday, 15:...	4s	⋮

# Entrenamiento en Edge Impulse

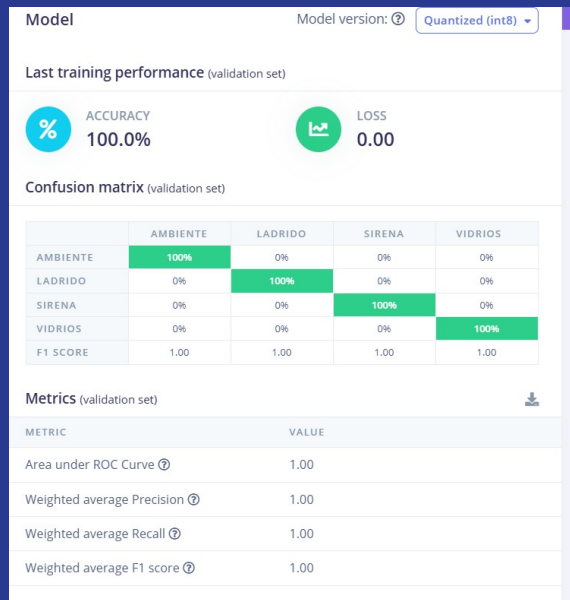
- Subida de datos
- Creación de impulso
- Clasificación
- Despliegue del modelo



# Evaluación del modelo


Se probaron muestras nuevas.


El modelo mostró buena precisión en clasificación.



# Implementación en Arduino


Se carga el modelo en el Arduino para clasificar sonidos en tiempo real

 Arduino library x



**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.



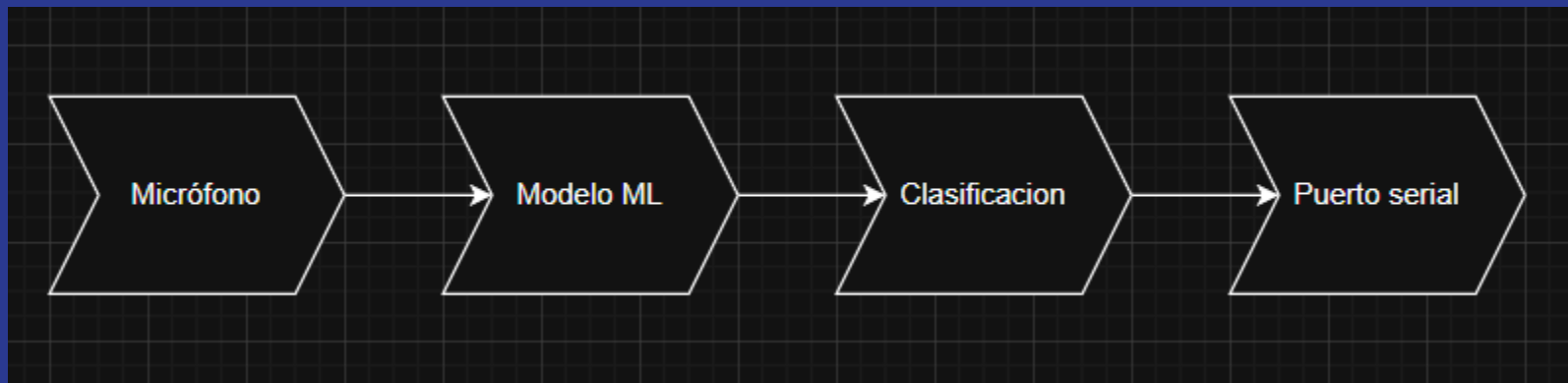
TensorFlow Lite

Quantized (int8)  
Selected ✓

	MFCC	CLASSIFIER	TOTAL
LATENCY	154 ms.	3 ms.	157 ms.
RAM	15.4K	6.4K	15.4K
FLASH	-	51.5K	-
ACCURACY			-

# Inferencia en tiempo real

El micrófono captura, el modelo infiere y se imprime la predicción por Serial.



# Validación funcional

Se hacen pruebas con sonidos reales. El sistema responde correctamente.

```
Predictions (DSP: 308 ms., Classification: 8 ms., Anomaly: 0 ms.):  
  ambiente: 0.96094  
  ladrido: 0.00000  
  sirena: 0.00000  
  vidrios: 0.03906
```

```
Predictions (DSP: 309 ms., Classification: 8 ms., Anomaly: 0 ms.):  
  ambiente: 0.00000  
  ladrido: 0.00000  
  sirena: 0.00000  
  vidrios: 0.99609
```

# Ventajas del sistema

Bajo costo, autónomo, no depende de red ni nube. Ideal para entornos locales.





# Limitaciones

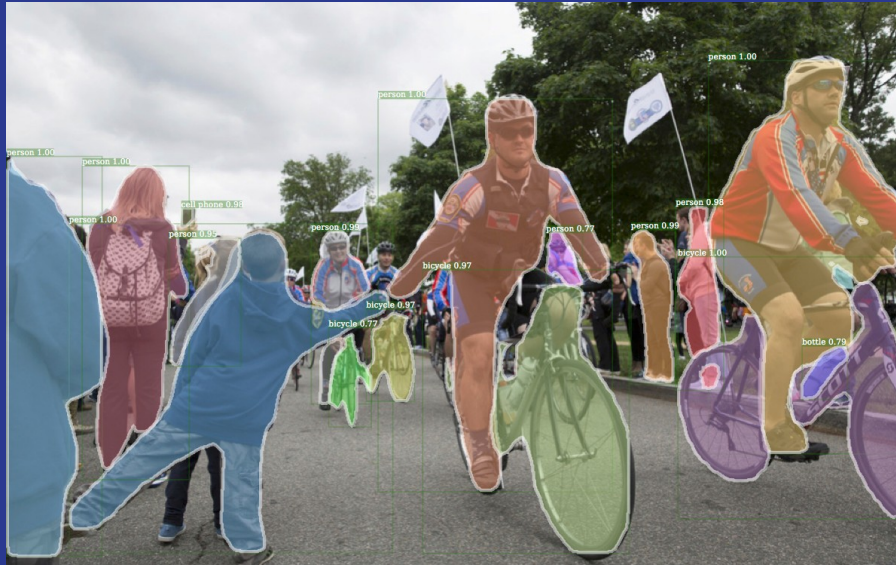
Solo clasificacion de sonido.



**No Image Available**

# Posibles mejoras

Agregar IoT, cámara o sensores de presencia. Expandir la base de sonidos.



# Conclusiones

Se logró clasificar sonidos relevantes con buena precisión en un sistema embebido.





# Fin

Muchas Gracias !!!