



**TECHNOLOGIE
STIFTUNG
BERLIN**



CityLAB Berlin



**TECHNISCHE
UNIVERSITÄT
BERLIN**

Offene Werkstatt: Du verstehst mich nie – ich probier's mal mit KI!

Mit Arduino Nano 33 BLE Sense

Reni Amalia Safitri
Sara Reichert

31. März 2023

1 Einleitung

In der offenen Werkstatt „Du verstehst mich nie – ich probier’s mal mit KI!“ wird der einfache Einstieg in die Welt von Tiny Machine Learning (TinyML) mit der Verwendung von Mikrocontroller Arduino Nano 33 BLE Sense vorgestellt. Als Beispiel werden hier die einzelnen Schritte gezeigt, wie man mit Spracheingabe die LEDs des Mikrocontrollers steuern kann. Das Machine Learning-Modell für die Spracherkennung wird mit der Plattform [Edge Impulse](#) erstellt. Das Ziel dieser Werkstatt ist es, dass die Teilnehmer:innen durch Experimentieren herausfinden, was genau mit TinyML möglich ist und was für Einschränkungen es gibt. Hierfür werden benötigt:

- 1 x Arduino Nano 33 BLE Sense
- 1 x Micro-USB-Kabel
- 1 x PC / Laptop mit USB Buchse

Alle benötigten Dateien können Sie [aus diesem GitHub-Repository](#) herunterladen.

2 LED Steuerung durch Spracheingabe

2.1 Arduino Einrichtung

1. Installation von Arduino IDE 2
In dieser Anleitung wird die Version 2.0.4 verwendet. Eine Anleitung zum Herunterladen und Installieren findet man [hier](#). Alternativ kann auch der [Arduino Web Editor](#) verwendet werden. Zur Einrichtung des Web-Editors findet man die Anleitung [hier](#).
2. Installation von Mbed OS Core für das Board
Eine Anleitung hierfür findet man [hier](#).
3. Installation von PDM Library
Weil wir für die Spracherkennung das Mikrofon verwenden werden, brauchen wir die [PDM Library](#). PDM steht für Pulse Density Modulation (deutsch: Pulsdichtemodulation) und ist eine Art Modulation, um analoge Signale im digitalen Bereich darzustellen. Sie ist oft in digitalen Mikrofonen zu finden. Eine Anleitung wie man in Arduino IDE eine neue Bibliothek installieren findet man [hier](#).
4. Installation von Arduino CLI
Eine Anleitung hierfür findet man [hier](#).

2.2 Edge Impulse Einrichtung

1. Installation von GNU Screen bei Linux
via `sudo apt install screen`.

2. Installation von [Python 3](#) und [Node.js](#)
Addition für Node.js bei Linux:

```
npm config get prefix
mkdir ~/.npm-global
npm config set prefix '~/.npm-global'
echo 'export PATH=~/.npm-global/bin:$PATH' >> ~/.profile
```

und bei MacOS:

```
npm config get prefix
mkdir ~/.npm-global
npm config set prefix '~/.npm-global'
echo 'export PATH=~/.npm-global/bin:$PATH' >> ~/.zprofile
```

3. Registrierung
Erstellen Sie ein Konto bei [Edge Impulse](#).
4. Erstellen des Projekts
Nach einer erfolgreichen Registrierung kann man direkt ein neues Projekt erstellen. Klicken Sie auf der rot markierten Schaltfläche.

Sign up successful!


Thanks **Renil**!

You have successfully signed up for Edge
Impulse.


 **Click here to build your first ML model!**


[Re-send activation email](#)


Beim Anlegen eines neuen Projekts wird meistens ein Wizard angezeigt, um den Typ der Daten festzulegen. Bitte wählen Sie „I know what I’m doing, hide this wizard!“ aus.


 Welcome to your new Edge Impulse project! ×

You're ready to add real intelligence to your edge devices. Let's set up your project. What type of data are you dealing with?

**Accelerometer data**
Analyze movement of your device in real-time to predict machine failure, detect human gestures, or monitor rotating machines.

**Audio**
Listen to what's happening around you to create voice interfaces, listen to keywords, detect audible events, or to hear what's happening around your device.

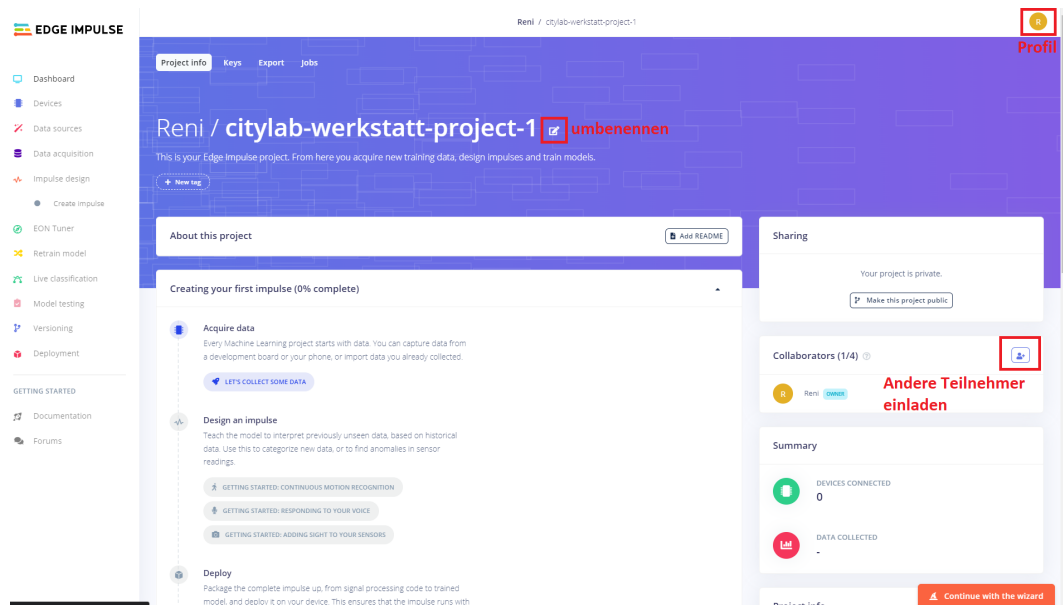
**Images**
Add sight to your sensors with image classification or object detection - to detect humans and animals, monitor production lines or track objects.

**Something else**
Different sensor? No problem! You can collect and import data from any sensor, from environmental sensors to radars - and deploy your trained model back to virtually any device.

I know what I'm doing, hide this wizard!

5. Projekteinstellung

Bei Bedarf können Sie das Projekt anpassen, z. B. umbenennen oder andere Teilnehmer:innen einladen. Der Vorteil von einem geteilten Projekt ist es, dass die Daten mit großen Variationen der Prosodie schneller gesammelt werden können. Unter Ihrem Profil können Sie noch weitere Projekte anlegen oder zwischen den Projekten wechseln.



6. Installation von Edge Impulse CLI

via `npm install -g edge-impulse-cli --force`.

7. Herunterladen von Edge Impulse Firmware Flash-Skript

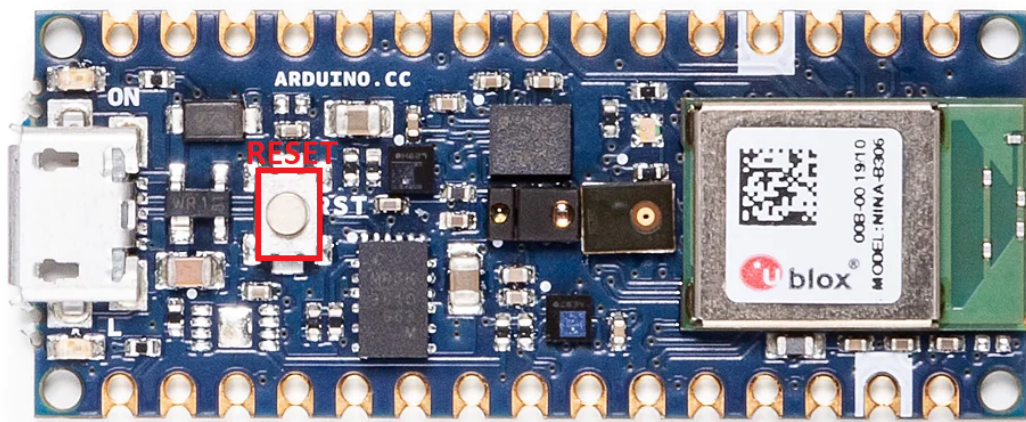
Die Datei können Sie [hier](#) herunterladen.

2.3 Edge Impulse mit dem Board verbinden

1. Mikrocontroller mit dem PC über das Kabel verbinden

2. Bootloader starten

Sie können den Bootloader starten, indem Sie zweimal die RESET-Taste drücken, danach soll die integrierte LED anfangen zu pulsieren.



3. Firmware flashen

Öffnen Sie das heruntergeladene Flash-Skript je nach Betriebssystem, um die neue Firmware zu flashen.

```
C:\WINDOWS\system32\cmd.exe

Eine neue Version von Arduino CLI ist verfügbar: 0.29.0 → 0.31.0
https://arduino.github.io/arduino-cli/latest/installation/#latest-packages
You're using an untested version of Arduino CLI, this might cause issues (found: 0.29.0, expected: 0.18.x)
Finding Arduino Mbed core...
arduino:mbed_nano 4.0.2      4.0.2  Arduino Mbed OS Nano Boards
Finding Arduino Mbed core OK
Finding Arduino Nano 33 BLE...
Finding Arduino Nano 33 BLE OK at COM6
arduino:mbed_nano 4.0.2      4.0.2  Arduino Mbed OS Nano Boards
Device       : nRF52840-QIAA
Version      : Arduino Bootloader (SAM-BA extended) 2.0 [Arduino:IKXYZ]
Address      : 0x0
Pages       : 256
Page Size   : 4096 bytes
Total Size  : 1024KB
Planes      : 1
Lock Regions: 0
Locked      : none
Security    : false
Erase flash

Done in 0.001 seconds
Write 352560 bytes to flash (87 pages)
[=====] 100% (87/87 pages)
Done in 15.700 seconds
Flashed your Arduino Nano 33 BLE development board
To set up your development with Edge Impulse, run 'edge-impulse-daemon'
To run your impulse on your development board, run 'edge-impulse-run-impulse'
Drücken Sie eine beliebige Taste . . .
```

Anschließend drücken Sie einmal die Reset-Taste, damit die neue Firmware gestartet werden kann.

4. Verbindung herstellen

via `edge-impulse-daemon --clean`. Danach werden Sie gefragt, die E-Mail-Adresse oder der Benutzername, das Passwort, ggf. der Projektname falls es mehrere gibt und der Name für das Board einzugeben.

```
Administrator: C:\WINDOWS\system32\cmd.exe - "node" "C:\Users\vasaf\AppData\Roaming\npm\node_modules\edge-impulse-...
Microsoft Windows [Version 10.0.19044.2728]
(c) Microsoft Corporation. Alle Rechte vorbehalten.

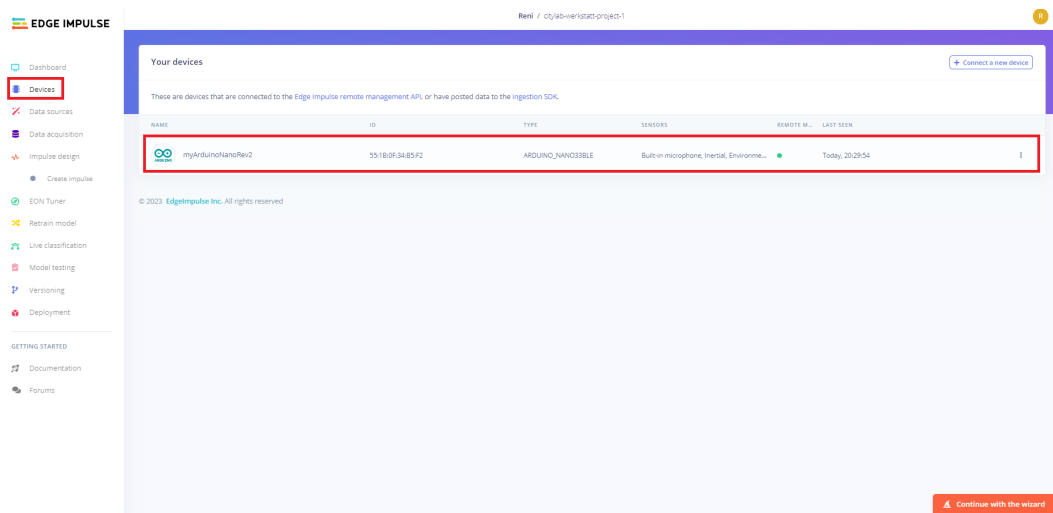
C:\WINDOWS\system32>edge-impulse-daemon --clean
Edge Impulse serial daemon v1.17.1
? What is your user name or e-mail address (edgeimpulse.com)? rsafitri@outlook.de
? What is your password? [hidden]
Endpoints:
  Websocket: wss://remote-mgmt.edgeimpulse.com
  API:       https://studio.edgeimpulse.com
  Ingestion: https://ingestion.edgeimpulse.com

[SER] Connecting to COM4
[SER] Serial is connected, trying to read config...
[SER] Clearing configuration
[SER] Clearing configuration OK
[SER] Failed to parse snapshot line [ ]
[SER] Retrieved configuration
[SER] Device is running AT command version 1.7.0

? To which project do you want to connect this device? Reni / citylab-werkstatt-project-1
Setting upload host in device... OK
Configuring remote management settings... OK
Configuring API key in device... OK
Configuring HMAC key in device... OK
[SER] Failed to parse snapshot line [ ]
[SER] Failed to parse snapshot line [ ]
[WS] Device is not connected to remote management API, will use daemon
[WS] Connecting to wss://remote-mgmt.edgeimpulse.com
[WS] Connected to wss://remote-mgmt.edgeimpulse.com
? What name do you want to give this device? myArduinoNanoRev2
[WS] Device "myArduinoNanoRev2" is now connected to project "citylab-werkstatt-project-1"
[WS] Go to https://studio.edgeimpulse.com/studio/203806/acquisition/training to build your machine learning model!
```

5. Überprüfen

Wenn der Prozess erfolgreich abgeschlossen ist, **bitte lassen sie das Fenster offen**. Dann ist das hinzugefügte Board unter Devices mit einem grünen Punkt sichtbar.



2.4 Erstellen des ML-Modells

Mit dem ML-Modell können Wörter von der Spracheingabe erkannt werden. Welche Wörter Sie verwenden wollen, ist Ihnen selbst überlassen. In diesem Beispiel werden 3 LEDs (RGB) gesteuert, also brauchen wir mindestens 4 Klassen / Labels (grün, blau, rot und aus), die erkannt werden sollen. Wichtig ist aber, dass die Erkennung von einem kurzen Wort schwieriger ist als von etwas längeren Wörtern (z. B. an und Bahn vs. Licht an und Licht aus). Ein guter Richtwert ist mindestens ein dreisilbiges Wort. Hier kann die Leistung der Spracherkennung nicht mit Google, Alexa und Siri verglichen werden.

1. Trainings- und Testdaten sammeln

Zuerst sammeln wir die Trainingsdaten. Bevor Sie mit dem Sampling beginnen, stellen Sie bitte erstmal die Parameter wie hier gezeigt. Sie können auch kürzere *Sampling length* auswählen, dann dauert das schneller, weil die Audiodatei dann auch kürzer ist. Sagen Sie während des Samplings Ihre Wörter in unterschiedliche Variationen mehrmals mit Pause dazwischen.

The screenshot displays the Edge Impulse web interface. On the left is a sidebar with navigation links. The main content area is titled 'Training data' and includes a 'Collected data' table with columns for sample name, label, added date, and length. A 'Record new data' form is highlighted with a red box, showing settings for device, label, sample length, sensor, and frequency. Below the form is an audio waveform visualization for a sample named 'gruen an.3smjdt2v'.

Nachdem die Audiodatei aufgenommen wurde, können Sie die Datei unter den 3 Punkten neben der Datei > Split Samples in mehrere Samples aufsplitten. Bitte wählen Sie das Fenster der Segmente so aus, dass die genau nur ihre Wörter haben ohne Pause.



Dieser Vorgang wiederholen Sie so lange, bis genug Samples für alle nötigen Wörter gesammelt sind. Je mehr Samples mit Prosodie-Variationen, umso genauer ist das Ergebnis. Bitte achten Sie auch darauf, dass es genug Testdaten gibt, Edge Impulse empfiehlt 80 % Trainingsdaten und 20 % Testdaten für alle Labels. Dafür kann man aber auch die Funktion „Perform train / test split“ verwenden.

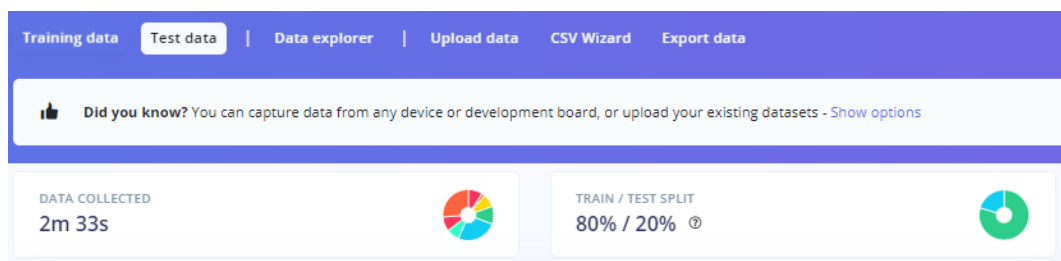
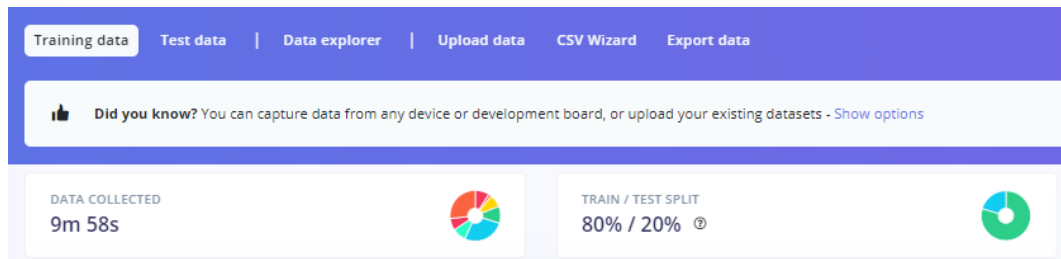
The screenshot shows the Edge Impulse web interface. The top navigation bar includes links for Training data, Test data, Data explorer, Upload data, CSV Wizard, and Export data. A sidebar on the left lists various tools like Dashboard, Devices, Data sources, Data acquisition, Impulse design, EON Tuner, Retrain model, Live classification, Model testing, Performance calibration, Versioning, and Deployment. The main content area displays 'DATA COLLECTED 6m 15s' and a 'TRAIN / TEST S...' status showing '100% / ...'. A modal window titled 'Dataset train / test split ratio' is open, providing instructions on data splitting and showing a suggested 80% / 20% split. Below this, it lists labels in the dataset (BACKGROUND NOISE, BLUE, GREEN, OFF, RED) with their respective 100% / 0% ratios and durations. A 'Perform train / test split' button is highlighted at the bottom of the modal.

Alternativ bietet Edge Impulse auch die Möglichkeit an, die Audiodateien hochzuladen.

2. Background Noise hinzufügen

Mit dem gleichen Prozess können wir jetzt Background Noise (z. B. Gespräch, Stille, Fernsehgeräusch usw.) als ein neues Label hinzufügen, weil das Modell nur das kennt, was es über die Daten gelernt hat. Aufgesplittet wird hier z. B. einfach jede 2 Sekunden. Alternativ kann

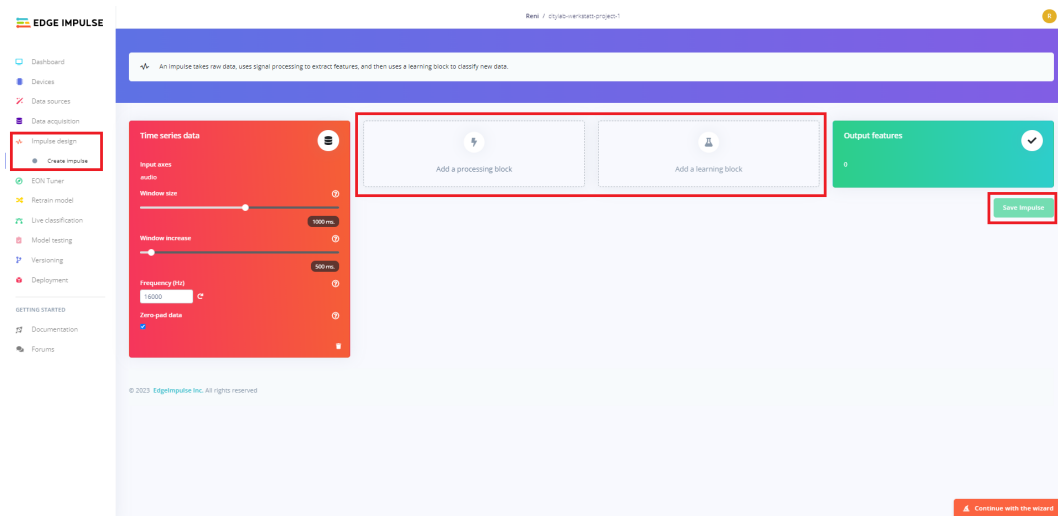
man den Wizard von Edge Impulse verwenden, um alle Background Noises automatisch zu erstellen. Am Ende sollen die Statistiken der Daten ungefähr so aussehen



3. Impulse erstellen

Der Impuls bestimmt wie das ML-Modell trainiert werden soll. Hierfür benötigt man einen Processing-Block, der Feature-Extraktion ausführt, und einen Learning-Block, der ein Muster von den Features erkennt. Danach können Sie den Impuls speichern.

Der Vorteil an Edge Impulse ist es, dass die Blöcke schon bereitgestellt sind und die Defaultwerte sind optimal, um gute Ergebnisse zu erhalten. Also benötigt man auch kaum Kenntnisse über Sprachsignalverarbeitung und Machine Learning.



Vorgeschlagen sind für Spracherkennung Audio (MFCC) als Processing-Block und Classification (Keras) als Learning-Block. Testen Sie das gerne aus, um herauszufinden, was für einen Einfluss die Änderung und Kombination von diesen Blöcken hat. In diesem Beispiel werden aber Audio (MFE) und Transfer Learning (Keyword Spotting) wie vom Wizard vorgeschlagen verwendet. Auch die Werte der Parameter sind unverändert.

⚡ Add a processing block



Did you know? You can [bring your own DSP code](#).

DESCRIPTION	AUTHOR	RECOMMENDED
Audio (MFCC) Extracts features from audio signals using Mel Frequency Cepstral Coefficients, great for human voice.	Edge Impulse	<button>Add</button>
Audio (MFE) Extracts a spectrogram from audio signals using Mel-filterbank energy features, great for non-voice audio.	Edge Impulse	<button>Add</button>
Spectrogram Extracts a spectrogram from audio or sensor data, great for non-voice audio or data with continuous frequencies.	Edge Impulse	<button>Add</button>
Audio (Syntiant) Syntiant only. Compute log Mel-filterbank energy features from an audio signal.	Syntiant	<button>Add</button>
Raw Data Use data without pre-processing. Useful if you want to use deep learning to learn features.	Edge Impulse	<button>Add</button>

Some processing blocks have been hidden based on the data in your project. [Show all blocks anyway](#)

[Add custom block](#)

Cancel

Add a learning block

×

Did you know? You can [bring your own model](#) in PyTorch, Keras or scikit-learn.

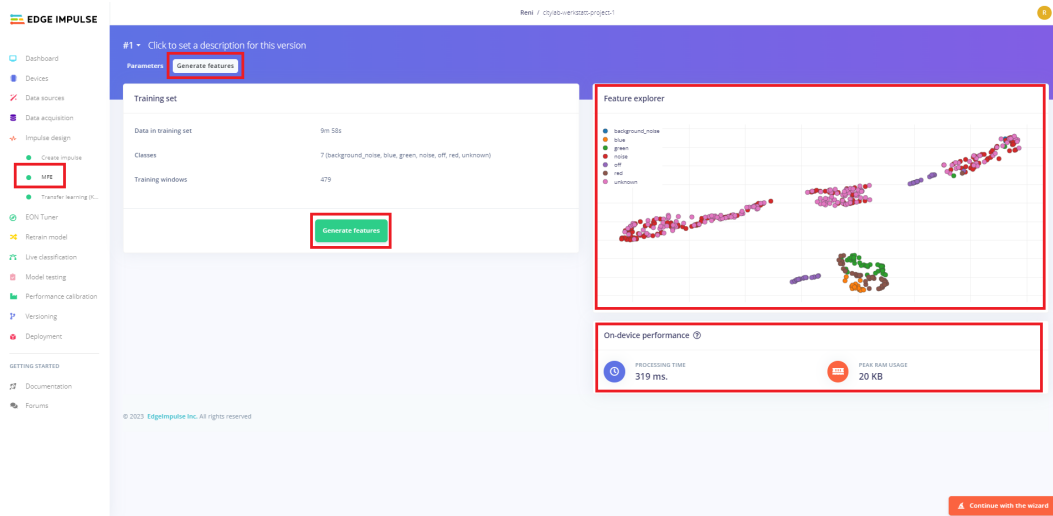
DESCRIPTION	AUTHOR	RECOMMENDED	
Classification Learns patterns from data, and can apply these to new data. Great for categorizing movement or recognizing audio.	Edge Impulse	★	<button>Add</button>
Regression Learns patterns from data, and can apply these to new data. Great for predicting numeric continuous values.	Edge Impulse		<button>Add</button>
Transfer Learning (Keyword Spotting) Fine tune a pre-trained keyword spotting model on your data. Good performance even with relatively small keyword datasets.	Edge Impulse		<button>Add</button>
Anomaly Detection (K-means) Find outliers in new data. Good for recognizing unknown states, and to complement classifiers. Works best with low dimensionality features like the output of the spectral features block.	Edge Impulse		<button>Add</button>
Classification (Keras) - BrainChip Akida™ Learns patterns from data, and can apply these to new data. Great for categorizing movement or recognizing audio.	BrainChip		<button>Add</button>

Some learning blocks have been hidden based on the data in your project. [Show all blocks anyway](#)

Cancel

4. Features generieren

Mit dem Processing-Block können dann die Features für die Training extrahiert werden. Hier sieht man dann auch die Schätzung der Leistung am Board.



5. Training starten

Mit dem Learning-Block wird das Training anhand der Features ausgeführt. Als Ergebnis sieht man einige Statistiken, die beschreiben, wie gut das Modell den Validierungsprozess geleistet hat. Die Werte sollen für eine sehr hohe Genauigkeit auch dementsprechend hoch sein. Falls die Werte zu niedrig sind, müssen die Daten nochmal angepasst bzw. vergrößert sein. Dabei ist es sehr wichtig, das richtige Board als Target einzustellen.

EDGE IMPULSE

Reni / citylab-werkstatt-project-1

#1 Click to set a description for this version

Target: Arduino Nano 33 BLE Sense (Cortex-M4F 64MHz)

Neural Network settings

Training settings

Number of training cycles 100

Learning rate 0.01

Validation set size 50 %

Auto-balance dataset ☒

Neural network architecture

Input layer (3,960 features)

MobileNetV2 0.35 (no final dense layer, 0.1 dropout)

Choose a different model

Output layer (7 classes)

Start training

Training output

Model

Model version: Quantized (int8)

Last training performance (validation set)

ACCURACY 96.7%

LOSS 0.28

Confusion matrix (validation set)

	BACKGR	BLUE	GREEN	NOISE	OFF	RED	UNKNOWN
BACKGR	100%	0%	0%	0%	0%	0%	0%
BLUE	0%	100%	0%	0%	0%	0%	0%
GREEN	0%	0%	95%	5%	0%	0%	0%
NOISE	4.9%	0%	0%	95.1%	0%	0%	0%
OFF	0%	0%	0%	0%	100%	0%	0%
RED	0%	0%	0%	0%	0%	100%	0%
UNKNOWN	1.1%	0%	0%	2.2%	0%	0%	96.6%
F1 SCORE	0.74	1.00	0.97	0.96	1.00	1.00	0.98

Data explorer (full training set)

On-device performance

INFERENC... 1.146 ms.

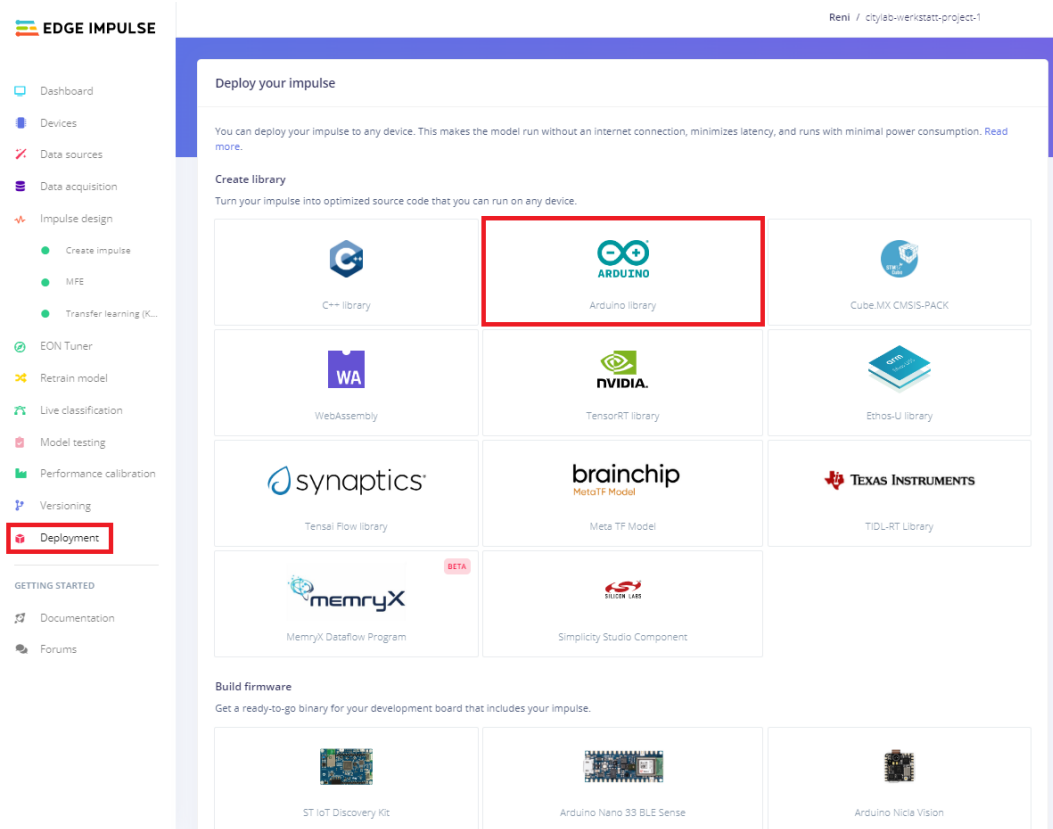
PEAK RAM ... 228,5K

FLASH USA... 580,6K

Bei der kostenlosen Version von Edge Impulse gibt es leider nur eine begrenzte Zeit von 20 Minuten für das Training. Falls das Training länger dauert, dann wird es abgebrochen. Man muss also entweder weniger Daten verwenden oder weniger Trainingszyklus einstellen.

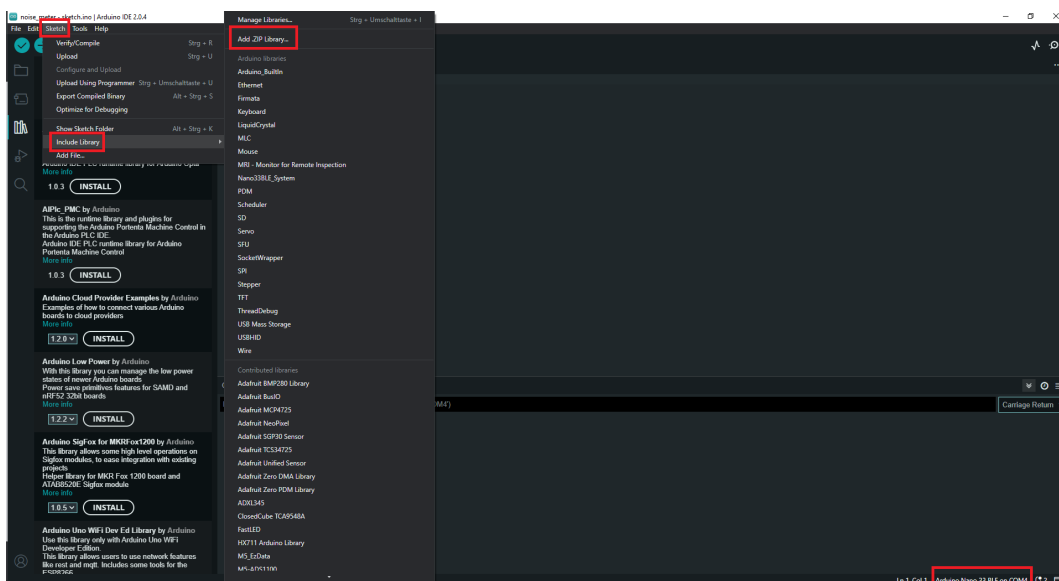
6. Deployment

Wenn man mit dem Ergebnis des Trainings zufrieden ist, kann das Modell dann als Library oder als Firmware exportiert werden. In diesem Beispiel wird das Modell als Arduino Library erstellt, damit wir es anpassen und verwenden können. Die Library wird dann als eine .zip-Datei gespeichert.



2.5 LED Steuerung mit dem ML-Modell

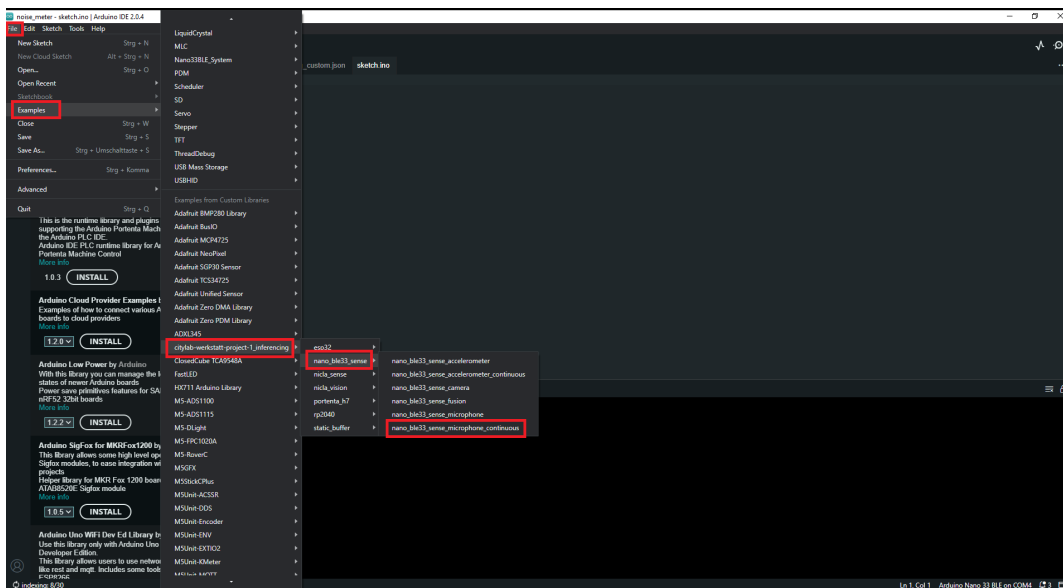
1. ML-Modell in Arduino IDE einbinden



Danach wählen Sie die gespeicherte Arduino Library, um diese einzubinden. Stellen Sie dabei auch sicher, dass der Board mit dem IDE verbunden ist.

2. Beispiel der Library anpassen

Das Beispiel der Library, was zu unserem Fall passt, findet man unter File > Examples > \$Projekt-Name\$ > nano_ble33_sense > nano_ble33_sense_microphone_continuous.



Wenn Sie diese Datei ins Board hochgeladen haben, dann können Sie die Klassifizierung der Spracheingabe in Serial Monitor einsehen. Danach können Sie das Code so erweitern, damit die LEDs je nach Klassifizierung gesteuert werden. Beispiel:

```

85  ...
86
87  /**
88   * @brief      Arduino main function. Runs the inferencing
89   *             loop.
90   */
91  void loop()
92  {
93      bool m = microphone_inference_record();
94      if (!m) {
95          ei_printf("ERR: Failed to record audio...\n");
96          return;
97      }

```

```

97
98     signal_t signal;
99     signal.total_length = EI_CLASSIFIER_SLICE_SIZE;
100     signal.get_data = &microphone_audio_signal_get_data;
101     ei_impulse_result_t result = {0};
102
103     EI_IMPULSE_ERROR r = run_classifier_continuous(&signal,
↪     &result, debug_nn);
104     if (r != EI_IMPULSE_OK) {
105         ei_printf("ERR: Failed to run classifier (%d)\n",
↪     r);
106         return;
107     }
108
109     if (++print_results >=
↪     (EI_CLASSIFIER_SLICES_PER_MODEL_WINDOW)) {
110         // print the predictions
111         ei_printf("Predictions ");
112         ei_printf("(DSP: %d ms., Classification: %d ms.,
↪     Anomaly: %d ms.)",
113         result.timing.dsp, result.timing.classification,
↪     result.timing.anomaly);
114         ei_printf(": \n");
115
116         // Hilfsvariablen für den hoechsten Wert
117         // Alternativ verwenden wir Threshold für die
↪     Klassifizierung
118         byte maxIndex = 0;
119         float maxValue =
↪     result.classification[maxIndex].value;
120
121         for (size_t ix = 0; ix < EI_CLASSIFIER_LABEL_COUNT;
↪     ix++) {
122             ei_printf("    %s: %.5f\n",
↪     result.classification[ix].label,
123             result.classification[ix].value);
124             // nach dem hoechsten Wert suchen
125             if(result.classification[ix].value > maxValue) {
126                 maxValue = result.classification[ix].value;
127                 maxIndex = ix;
128             }
129         }
130
131         // Der hoechste Wert ist die Klassifizierung,

```

```

132         // also je nach Label LED steuern
133         if(result.classification[maxIndex].label == "red"){
134             // label: red, also LED rot an, andere aus
135             digitalWrite(LED_R, LOW);
136             digitalWrite(LED_G, HIGH);
137             digitalWrite(LED_B, HIGH);
138         }
139         ↪ else if(result.classification[maxIndex].label ==
           "green"){
140             // label: green, also LED gruen an, andere aus
141             digitalWrite(LED_R, HIGH);
142             digitalWrite(LED_G, LOW);
143             digitalWrite(LED_B, HIGH);
144         }
145         ↪ else if(result.classification[maxIndex].label ==
           "blue"){
146             // label: blue, also LED blau an, andere aus
147             digitalWrite(LED_R, HIGH);
148             digitalWrite(LED_G, HIGH);
149             digitalWrite(LED_B, LOW);
150         }
151         ↪ else if(result.classification[maxIndex].label ==
           "off"){
152             // label: off, also LEDs aus
153             digitalWrite(LED_R, HIGH);
154             digitalWrite(LED_G, HIGH);
155             digitalWrite(LED_B, HIGH);
156         }
157
158
159         #if EI_CLASSIFIER_HAS_ANOMALY == 1
160         ↪ ei_printf("    anomaly score: %.3f\n",
           result.anomaly);
161         #endif
162
163         print_results = 0;
164     }
165 }
166
167 ...

```

3. Testen

Testen Sie bitte ob die Spracheingabe für die LED Steuerung mit Ihrem Board funktioniert! Beim ersten Hochladen dauert das etwas länger.

3 Weitere Projektideen mit Arduino Nano 33 BLE Sense

- intelligentes Roboterauto mit Hindernisausweichung
- per Bluetooth gesteuertes Auto
- Thermostat
- audiobasierte Überwachung (mit ML)
- ...
- ...
- ...
- ...
- ...

Mit anderen Boards von Arduino hat man dann auch unterschiedliche ggf. auch mehr Möglichkeiten. Z. B. mit Arduino Nano 33 IoT kann man sich über WLAN mit dem Arduino IoT Cloud verbinden, was wiederum auch per Handy-App kann überwacht werden. Oder mit dem Arduino Nano 33 BLE Sense Rev 2 hat man die Möglichkeit, die Spracheingabe direkt mit Arduino Speech Recognition Engine zu erfassen.