KI & Maschinenlernen auf dem Mikrocontroller

Teil II: Edge Impulse mit dem TinyML-Kit & ESP-Eye

Urheber: Thomas Jörg

Stand 20. März 2023

Abb. 1: Eigenes Foto Tiny ML Kit

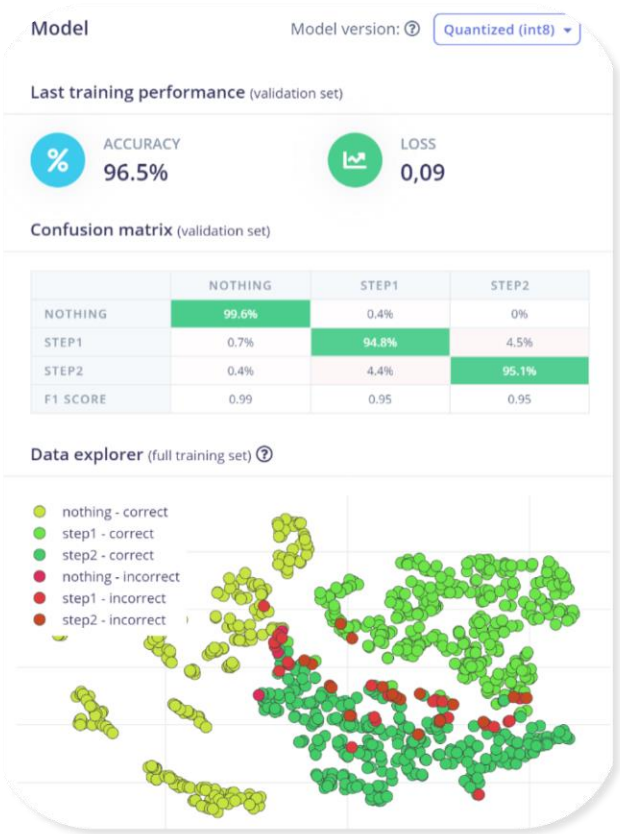


Abb. 3: Eigenes Screenshot von Edge Impulse IDE

Abb. 2: Eigenes Foto Esp-Eye (ESP32)

Inhaltsverzeichnis

Warum Edge Impulse?

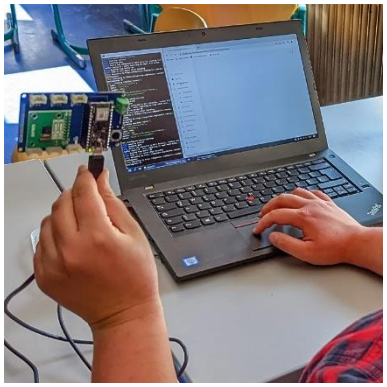


Abb. 4: Eigenes Foto aus dem Unterricht

"*Edge Impulse ist die führende Entwicklungsplattform für maschinelles Lernen, […] auf eingebetteten Geräten für Sensoren, Audio und Computer Vision in großem Umfang. Es ermöglicht den Einsatz von hochoptimierter maschineller Intelligenz auf Hardware, die von MCUs über CPUs […] reicht. Mit Edge Impulse erhalten Entwickler […] das Werkzeug, das sie benötigen, um ihre realen Probleme zu lösen.*"[1] (Übers. v. T. Jörg)

Oder in eigenen Worten: Der Prozess, um ML/Neuronale Netze auf Microcontroller zu portieren, ist komplex. Die Einstiegshürde, um zum Beispiel mit Tensorflow eigene Modelle zu trainieren ist sehr hoch. Die Kenntnis von sehr speziellen und schwierig zu erlernenden Bibliotheken und Workflows ist dazu notwendig.

Diesen Prozess kann man vereinfachen und „streamlinen": Edge Impulse verpackt den gesamten Prozess von

1. Datenaufnahme,
2. Aufbereitung,
3. Feature extraction,
4. Modellauswahl,
5. Hyperparameter-Tuning,
6. Training,
7. Test, Validierung und
8. Deployment

unter einem Dach. Das ist möglich, ohne eine einzige Codezeile zu schreiben.

Aber:

Kenntnisse der einzelnen Schritte sind notwendig. In der Feature-Extraction sind z.B. Kenntnisse von FFTs oder Powerspektren nötig, beim Erstellen von Neuronalen Netzen benötigt man Kenntnisse über grundsätzliche Netztopologien oder CNNs, in Test und Validierung sollte eine Confusion Matrix bekannt sein.

Hier liegt die didaktische Stärke von Edge Impulse:

> Man kann sich auf den Workflow und die eigentlichen Maschinenlern-Algorithmen konzentrieren. Müsste man alles beispielsweise in Python-Skripts mit skLearn oder Tensorflow programmieren, würde der Schüler höchstwahrscheinlich den Wald vor lauter Bäumen nicht mehr sehen.

[1] Quelle: https://www.arm.com/partners/ai-ecosystem-catalog/edge-impulse

Warum das TinyML-Kit (Arduino 33 BLE Sense)?

Das TinyML-Kit wurde von der Universität Harvard entwickelt als Modell- und Lehrsystem für Tensorflow Micro. Es handelt sich zwar einerseits nicht um die stärkste Hardware, aber die bei weitem am besten unterstützte! Diese Unterstützung führt dazu, dass man weniger Hürden und Schwierigkeiten überwinden muss. Außerdem gibt es sehr gute – auch kostenfreie – Kurse dazu, nämlich von der Universität Harvard und von Edge Impulse, die auf der Plattform Coursera angeboten werden.

Was hat das Board, bzw. das Kit zu bieten?

„Alles“.

(Etwas genauer: Alles, was man für die umfassende Einführung in das Thema ‚KI auf µC‘ benötigt)

https://docs.arduino.cc/tutorials/nano-33-ble-sense/cheat-sheet
https://docs.arduino.cc/hardware/nano-33-ble-sense
https://infocenter.nordicsemi.com/pdf/nRF52840_PS_v1.1.pdf

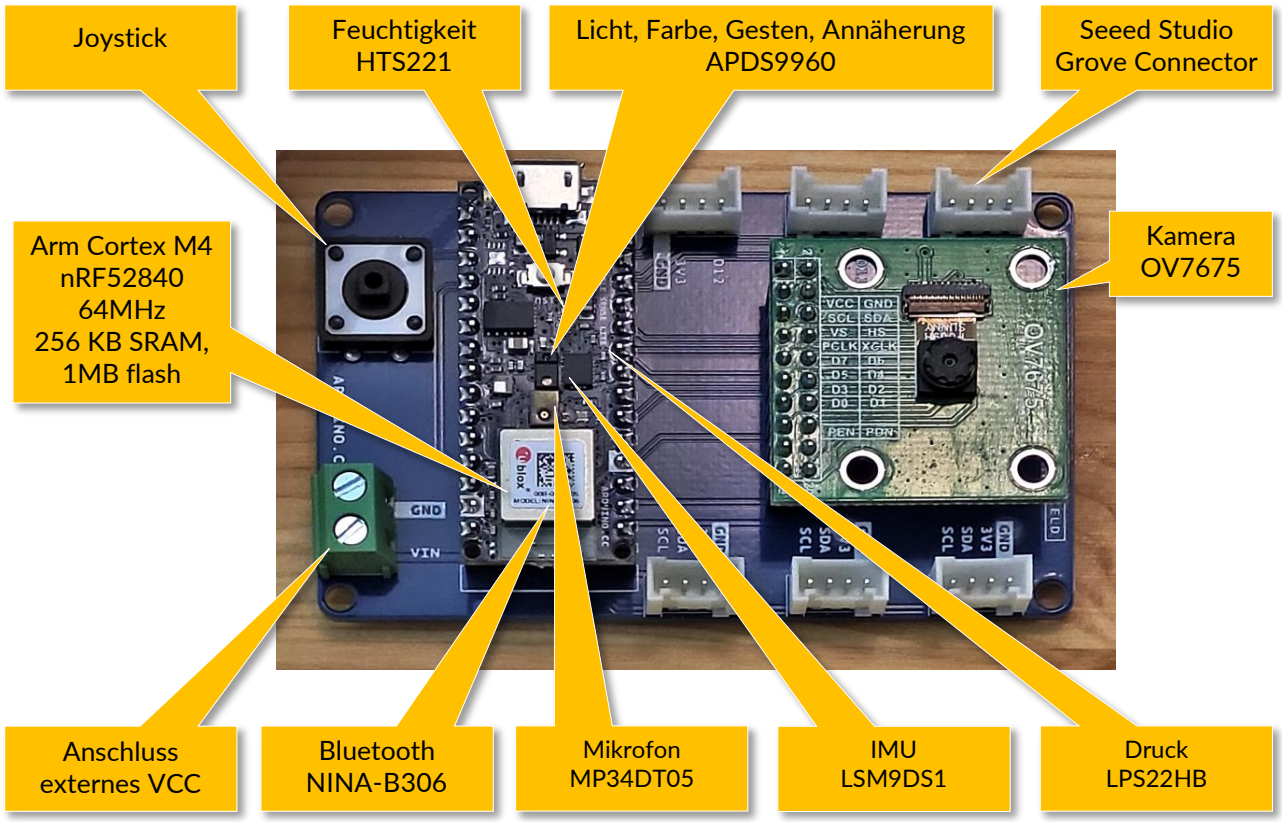


Abb. 5: Eigenes Foto & eigene Beschriftung.

Alternativ: Warum das ESP-Eye (ESP32)?

Abb. 6: Eigenes Foto ESP-Eye

Das ESP-Eye ist ein sehr leistungsfähiges Microcontrollerboard, basierend auf einem ESP32. Es ist ursprünglich dazu gedacht, sich direkt mit dem Internet zu verbinden und dort mit AWS (Amazon Web Service) zu kommunizieren. Auf den folgenden Seiten kann man sich detailliert über das Board informieren:

https://www.espressif.com/en/products/devkits/esp-eye/overview

https://devices.amazonaws.com/detail/a3G0h0000077i2JEAQ/ESP-EYE

Für den Aufbau der Kommunikation verfügt es über folgende Ausstattung:

Sensorik:

- 2MP-Kamera (OV 2640)
- Mikrophon

Hardware:

- 8 MB PSRAM (Hervorragend geeignet für größerer Neuronale Netze z.B. für Object Detection etc.)
- 4 MB flash
- WLAN-Antenne

Quelle:

https://github.com/espressif/esp-who/blob/master/docs/en/get-started/ESP-EYE_Getting_Started_Guide.md

Es ist damit zwar nicht annähernd so vielseitig wie das Harvard-TinyML-Kit, aber das, was es kann, ist enorm leistungsfähig.

Darüber hinaus wird es mittlerweile auch voll durch Edge-Impulse unterstützt. Das heißt, es wird

- Das Board ist von Edge Impulse gut dokumentiert
- Es existiert eine voll funktionsfähige Firmware
- Es bindet sich unmittelbar in die Entwicklungsumgebung ein
- Es ist voll lauffähiger Arduino-Quellcode herunterladbar
- Aufgrund der starken Hardware sind aufwändigere Modelle möglich

Einrichtung von Edge Impulse

Abb. 7: Edge-Impulse-Logo [RAIL v0.1]

Im folgenden Abschnitt wird die Einrichtung des Computers erläutert, den ein Schüler benutzen soll.

Das soll der Schüler bitte nicht selbst machen!

Empfehlung:

Es wird ein Rechner-Image vorbereitet, welches auf mehrere Rechner verteilt wird.

Grundinstallation der benötigten Software

In der aktuellen Version benötigt man lediglich drei Schritte:

1. NodeJS mit Chocolatey inkl. VisualStudio BuildTools und Python (getestet mit node-v18.13.0-x86, **32 Bit**)
2. Edge CLI: Spezielle Schnittstellen-Software von Edge-Impulse
3. Spezielle Ausstattung für den Arduino 33 BLE Sense (Arduino-CLI) oder das ESP-Eye (esptool)

1. Node.js inkl. Python und Visual Studio Build Tools

Mit der aktuellen Version von Node.js hat sich die Installation deutlich erleichtert - Node.js installiert im Hintergrund Python und die benötigten Komponenten der Visual Studio Build Tools mit. Damit erübrigt sich die zusätzliche Installation von Visual Studio und Python.

Abb. 8: NodeJS-Logo [Public Domain Mark 1.0]

Getestet wurde die aktuelle Installationsvariante mit Node. v18.13.0-x86.

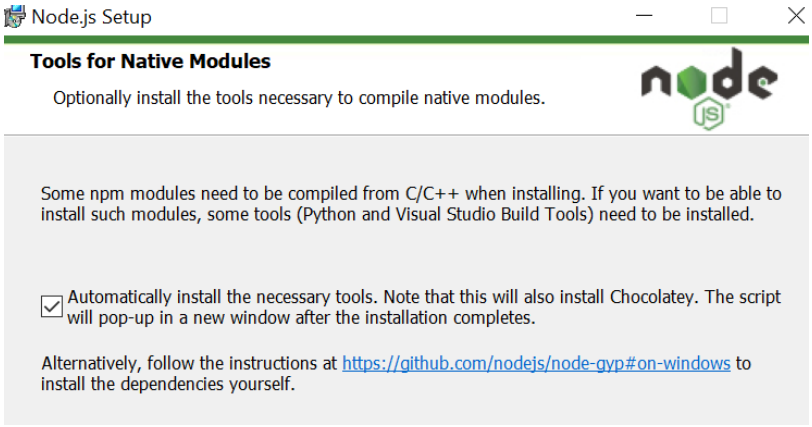


Abb. 9: Eigener Screenshot NodeJS-Installation

Bei der Installation wird **UNBEDINGT** das Häkchen bei

„*Automatically install the necessary tools. ...*“

angewählt.

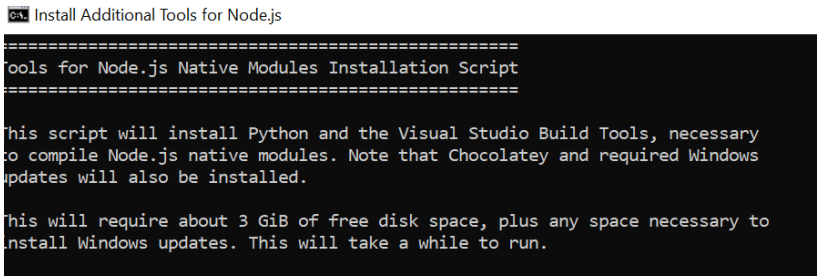


Abb. 10: Eigener Screenshot Kommandozeile

Der unterschied wird im nächsten Schritt sichtbar. Nachdem Node.js fertig installiert ist, geht das Konsolen-Fenster für die Installation von zusätzlichen Tools auf. Hier kann man dem Text entnehmen, dass **sowohl Python, als auch Visual Studio Build Tools per Script installiert werden**.

Die Installationschritte, bzw. den Installationsfortschritt kann man später im Powershell-Fenster nachvollziehen:

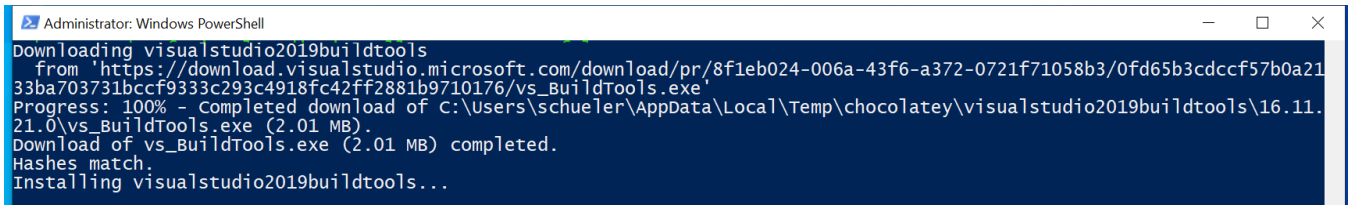


Abb. 11: Eigener Screenshot Powershell

Falls bei der Installation Fehler auftreten sollten, kann die Installation durch das Ausführen des Befehls vervollständigt bzw. fertiggestellt werden:

```
choco install visualstudio2019-workload-vctools
```

2. Edge CLI installieren

Aus der CMD-Konsole, aufgerufen mit Administrator-Rechten, wird CLI mit folgendem Befehl installiert:

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

Abb. 12: Edge-Impulse-Logo [RAIL v0.1]

npm install edge-impulse-cli

```
Microsoft Windows [Version 10.0.19044.1706]
(c) Microsoft Corporation. Alle Rechte vorbehalten.

C:\Windows\system32>npm install -g edge-impulse-cli
[                    ] \ idealTree:string-width: sill fetch manifest tslib@^1.9.0
```

Abb. 13: Eigener Screenshot Kommandozeile

Administrator: Eingabeaufforderung

```
Microsoft Windows [Version 10.0.19044.1706]
(c) Microsoft Corporation. Alle Rechte vorbehalten.

C:\Windows\system32>npm install -g edge-impulse-cli
npm WARN deprecated @zeit/dockerignore@0.0.5: "@zeit/dockerignore" is no longer maintained
npm WARN deprecated har-validator@5.1.5: this library is no longer supported
npm WARN deprecated request-promise@4.2.4: request-promise has been deprecated because it extends the r
uest package, see https://github.com/request/request/issues/3142
npm WARN deprecated debug@4.1.1: Debug versions >=3.2.0 <3.2.7 || >=4 <4.3.1 have a low-severity ReDos
sed in a Node.js environment. It is recommended you upgrade to 3.2.7 or 4.3.1. (https://github.com/visi
ues/797)
npm WARN deprecated debug@4.1.1: Debug versions >=3.2.0 <3.2.7 || >=4 <4.3.1 have a low-severity ReDos
sed in a Node.js environment. It is recommended you upgrade to 3.2.7 or 4.3.1. (https://github.com/visi
ues/797)
npm WARN deprecated debug@4.1.1: Debug versions >=3.2.0 <3.2.7 || >=4 <4.3.1 have a low-severity ReDos
sed in a Node.js environment. It is recommended you upgrade to 3.2.7 or 4.3.1. (https://github.com/visi
ues/797)
npm WARN deprecated uuid@3.4.0: Please upgrade  to version 7 or higher.  Older versions may use Math.ra
 circumstances, which is known to be problematic.  See https://v8.dev/blog/math-random for details.
npm WARN deprecated request@2.88.0: request has been deprecated, see https://github.com/request/request
```

Abb. 14: Eigener Screenshot Kommandozeile

Troubleshooting

Falls bei der Installation Fehler auftreten sollten, könnten weitere Schritte in Betracht gezogen werden:

4. Manuelle Installation von Python

Es wird eine Installation von Python benötigt. Nach der Installation ist manchmal die Einbindung von Python zu den Umgebungsvariablen gesondert auszuführen.

Dafür startet man die Installationsdatei erneut und wählt unter „Advanced Options“: „Add Python to environment variables“.

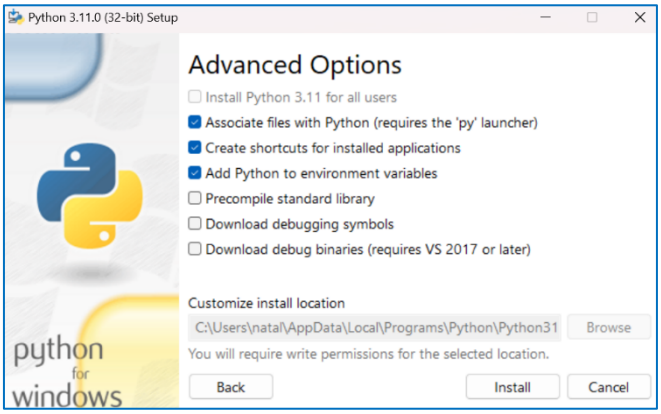


Abb. 15: Eigener Screenshot Python-Installation

5. Separate Installation von Visual Studio Build Tools[2] 2019

a) herunterladen

Das Paket kann unter folgendem Link heruntergeladen werden:

https://visualstudio.microsoft.com/de/thank-you-downloading-visual-studio/?sku=BuildTools&rel=16
(ca. 1,5MB große exe-Datei)

Abb. 16: Logo Visual Studio, [Public Domain Mark 1.0]

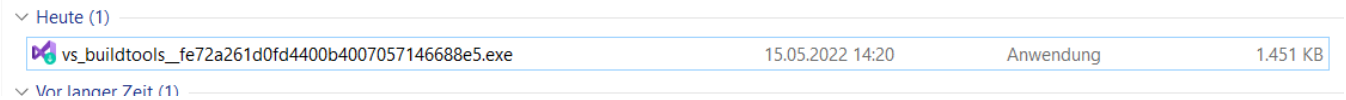


Abb. 17: Eigener Screenshot VS BuildTools-Download

Alternativ findet man es auch direkt über “visual studio download” -> “ältere Versionen” (https://visualstudio.microsoft.com/de/vs/older-downloads/).
Allerdings wird hier unter Umständen für das Herunterladen das Login benötigt.

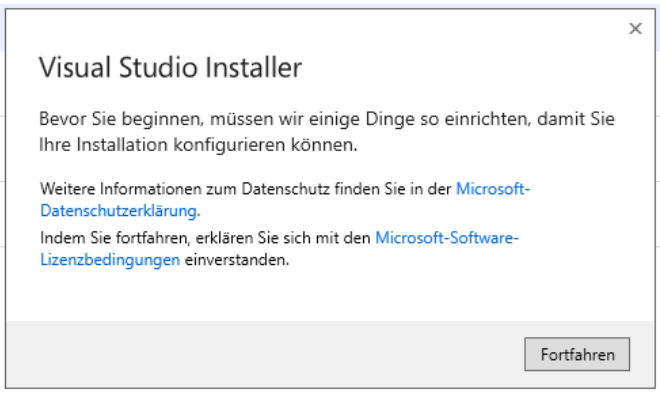


Abb. 18: Eigener Screenshot VS-BuildTools-Installation

[2] Es gibt zwar Alternativen zum Microsoft-Angebot, diese haben aber weder den gleichen Funktionsumfang noch sind sie mit der anderen verwendeten Software kompatibel.

b) Installation von “Visual Studio Build Tools 2019”

Es werden als zusätzlicher Workload „Desktopentwicklung mit C++“ benötigt.

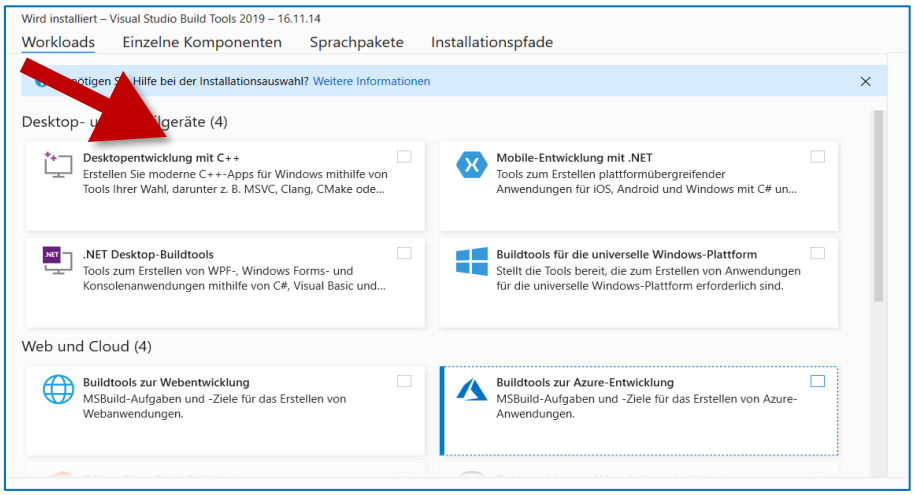


Abb. 19: Eigener Screenshot VS-BuildTools-Installation

Wenn es erfolgreich installiert wurde, kann man Installer beenden, **ohne Visual Studio zu starten**.

Abb. 20: Eigener Screenshot VS-BuildTools-Installation

3a. Installation für den Arduino 33 BLE Sense (Harvard Kit)

Arduino-CLI (Command Line Interface)

Die Original-Dokumentation des Herstellers findet man hier:
https://docs.edgeimpulse.com/docs/development-boards/arduino-nano-33-ble-sense

1. CLI (CLI steht für ‚Command Line Interface') ist für das Befehlszeilen-gestützte Hochladen der Programme auf den Arduino notwendig. Es kann von hier heruntergeladen werden:
   https://arduino.github.io/arduino-cli/0.21/installation/

   Achten Sie beim Auspacken auf den Pfad, man wird auf das Verzeichnis häufiger zugreifen müssen.

2. OPTIONAL, zum Upgraden der CLI, nicht unbedingt nötig:
   CMD-Konsole als Administrator öffnen und in das Verzeichnis mit der ausgepackten CLI wechseln, dort:

   ```
   arduino-cli.exe upgrade
   ```

   ausführen, um CLI zu installieren.

Latest release

Platform		
Linux	32 bit	64 bit
Linux ARM	32 bit	64 bit
Windows	32 bit	64 bit
macOS		64 bit

Abb. 21: Eigener Screenshot Arduino-Webseite

Edge-Firmware für Arduino installieren

Für das Arduino-Board wird die Edge-Firmware benötigt, damit das Board in Edge Impulse unter "Devices" erreichbar und sichtbar wird. Die Firmware kann von hier als Zip-Datei heruntergeladen werden:

https://cdn.edgeimpulse.com/firmware/arduino-nano-33-ble-sense.zip

Entpacken sollte man es direkt in den Ordner mit der Arduino-CLI, weil die Pfade in der .bat-Datei es so vorsehen:

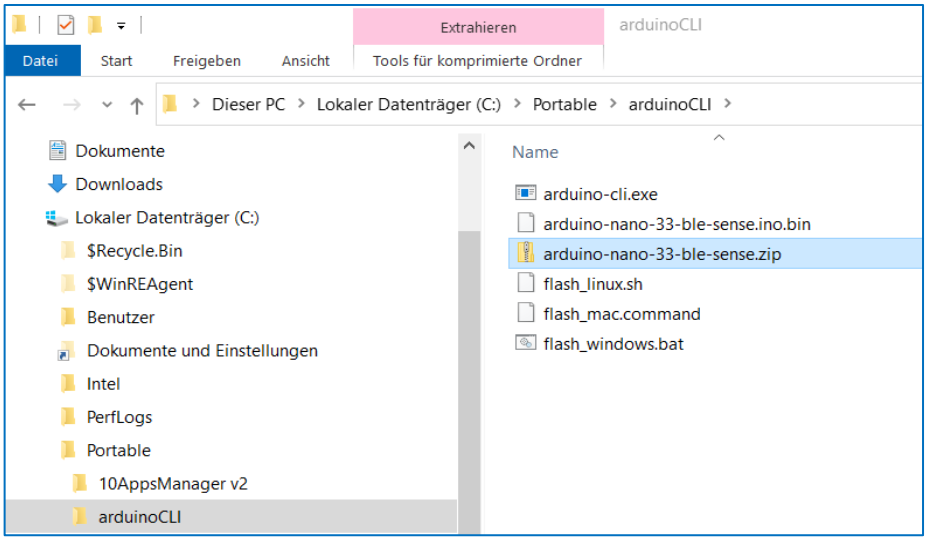


Abb. 22: Eigener Screenshot Windows-Explorer

Flashen kann man den Arduino-Mikrocontroller nun mit der Firmware (Achtung: Adminrechte erforderlich):

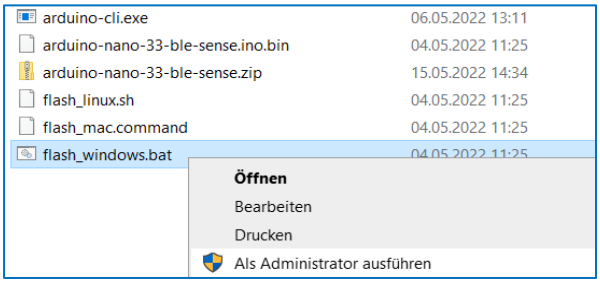


Abb. 23: Eigener Screenshot Windows-Explorer

- Arduino anschließen, Doppelklick auf “RST”
  (orange LED leuchtet dann)

- flash-windows.bat
  von der Edge-Firmware ausführen
  (Zugriff zulassen)

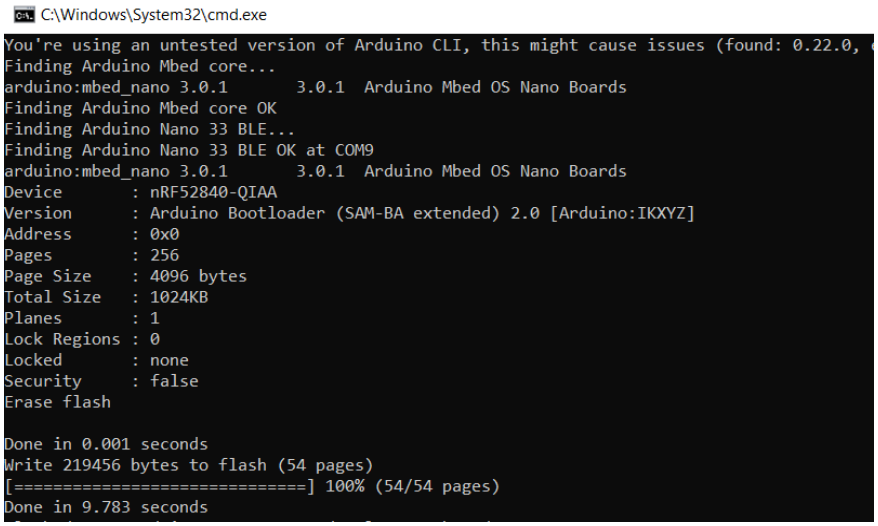


Abb. 24: Eigener Screenshot Kommandozeile

Anmerkung:
Diese Firmware-Installation wird vermutlich häufiger durchgeführt. Deshalb sollte das Arduino-CLI-Verzeichnis mit den Firmware-Dateien leicht erreichbar sein.

Außerdem kann man einen ‚abgeschossenen‘ Arduino Nano 33 BLE-Sense sehr gut und immer wieder mittels dieser Firmware-Installation wieder aufwecken, bzw. reparieren!

3b Installation für das ESP-Eye

Abb. 25: Eigenes Foto ESP-Eye (ESP32)

ESP-Tool und CP2102-Treiber

Die Original-Dokumentation des Herstellers findet man hier:

https://docs.edgeimpulse.com/docs/development-platforms/officially-supported-mcu-targets/espressif-esp32

1. ESP-Tool:

   Weil Node-JS mitsamt der Zusatz-Software bereits installiert ist, existiert auch eine Python-Installation, deren Pfad sich in den globalen Pfadeintragungen befindet. Deshalb *sollte* es funktionieren, in einem CMD-Fenster (als Administrator öffnen) folgenden Befehl einzugeben:

   ```
   pip install esptool
   ```

   Abb. 26: Eigener Screenshot Kommandozeile

   Quelle: https://pypi.org/project/esptool/

2. Danach schließt man das ESP-Eye mit dem Kabel an den Rechner an. Üblicherweise wird das Gerät von Windows nicht erkannt, weil die Treiber für die Kommunikation nicht vorhanden sind. Diese kann man von hier herunterladen: https://www.silabs.com/developers/usb-to-uart-bridge-vcp-drivers?tab=downloads

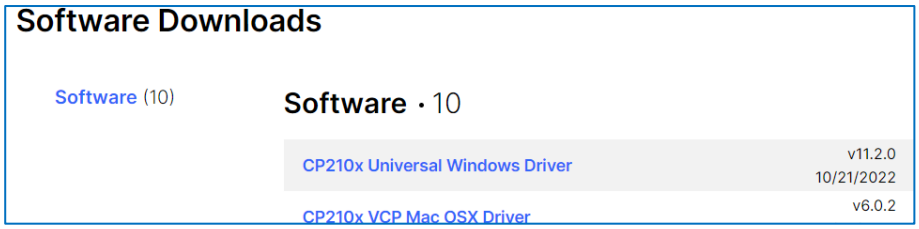
   

   Abb. 27: Eigener Screenshot SiliconLabs-Webseite

3. Den Geräte-Manager starten:

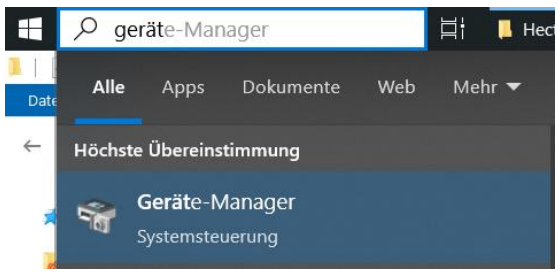
   

   Abb. 28: Eigener Screenshot Windows-Oberfläche

4. Im Gerätemanager den nicht-installierte ESP-Eye anwählen und manuelle Teiberinstallation starten: „Auf meinem Computer nach Treibern suchen“

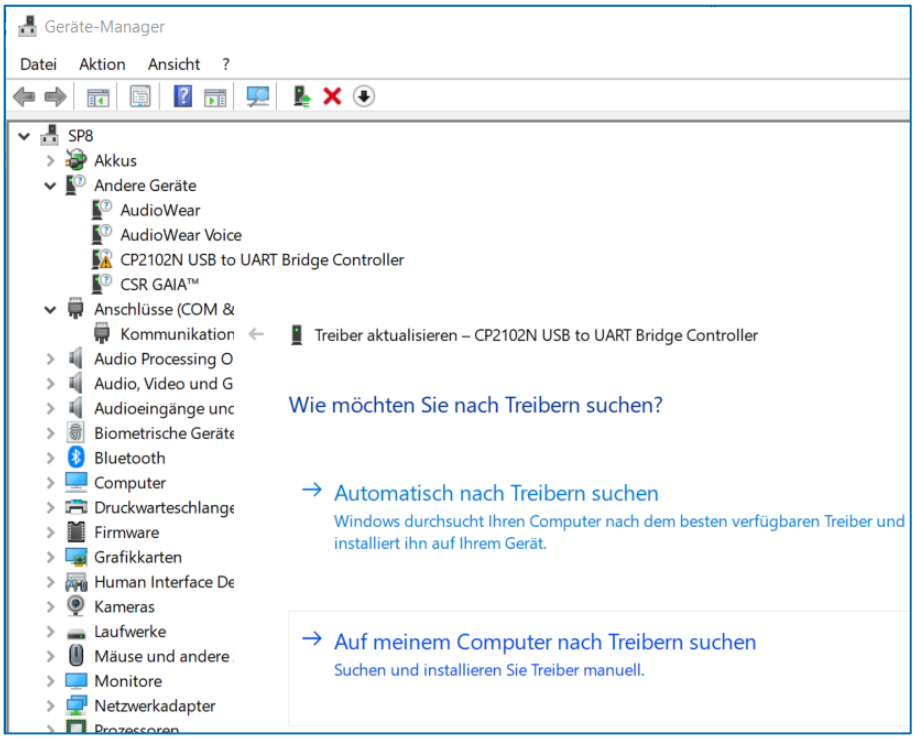


Abb. 29: Eigener Screenshot Windows-Oberfläche

5. Zuletzt sollte dann so aussehen:

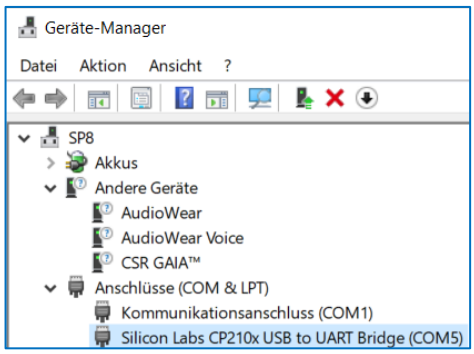


Abb. 30: Eigener Screenshot Windows-Oberfläche

Edge-Firmware für ESP-Eye installieren

Für das ESP-Eye wird die Edge-Firmware benötigt, damit das Board in Edge Impulse unter “Devices” erreichbar und sichtbar wird. Die Firmware kann von hier als Zip-Datei heruntergeladen werden:

https://cdn.edgeimpulse.com/firmware/espressif-esp32.zip

Entpacken sollte man es direkt in einen eigenen Ordner, zum Beispiel so:

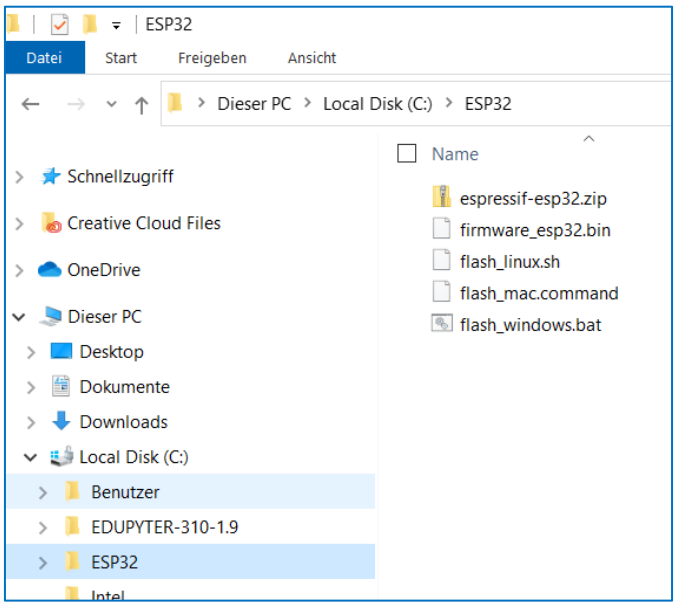


Abb. 31: Eigener Screenshot Windows-Explorer

Flashen kann man den Arduino-Mikrocontroller nun mit der Firmware (Achtung: Adminrechte erforderlich):

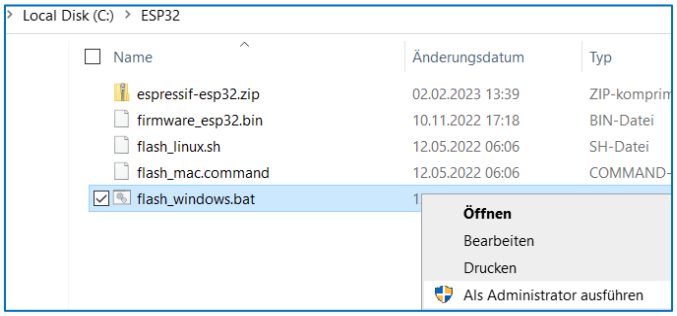


Abb. 32: Eigener Screenshot Windows-Explorer

flash-windows.bat von der Edge-Firmware ausführen

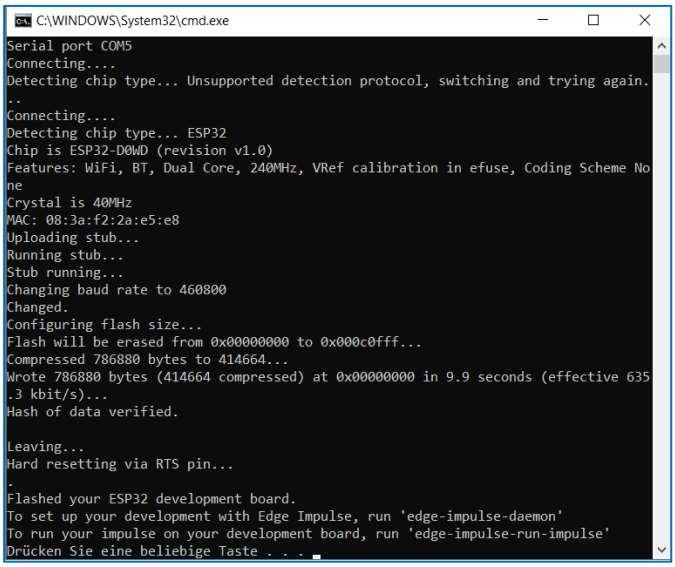


Abb. 33: Eigener Screenshot Kommandozeile

Das erste Projekt

Registrierung

Das Nutzen dieser Entwicklungsplattform benötigt eine Registrierung, wobei nur eine gültige Emailadresse benötigt wird.

https://studio.edgeimpulse.com/signup

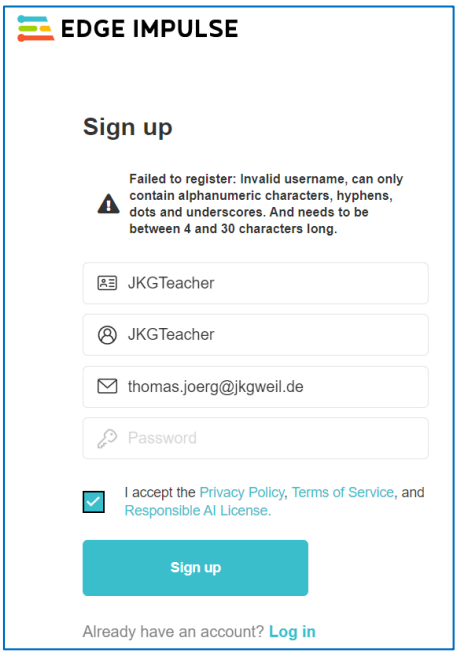


Abb. 34: Eigener Screenshot Edge-Impulse-Webseite

Tipp: Es reicht zuerst, wenn nur der Lehrer sich registriert. Mit den zur Verfügung gestellten Zugangsdaten kann jeder Schüler sich ein eigenes Projekt damit anlegen.

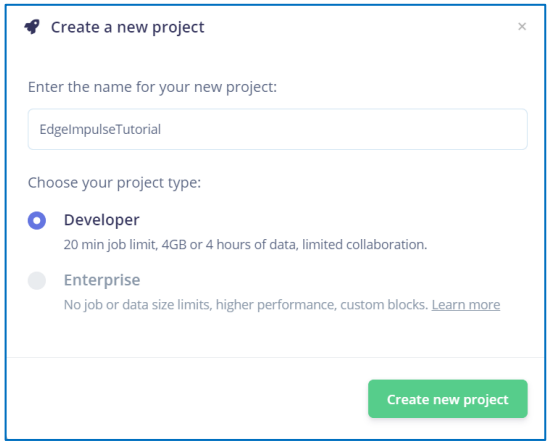


Abb. 35: Eigener Screenshot Edge-Impulse-Projekterstellung

Übersicht über die einzelnen Prozessschritte

Die folgenden Schritte sind für fast alle Aufgabenstellungen identisch. Der Workflow entspricht dabei exakt der klassischen Vorgehensweise bei der Programmierung von Python/Tensorflow. Im Prinzip kann man sich EdgeImpulse als grafische Benutzeroberfläche für den Python-Workflow vorstellen.

Abb. 36: Selbst erstellte Grafik

Neues Projekt starten & Device einbinden

Man beginnt in Edge Impulse mit dem Anlegen eines neuen Projekts.

Dabei kann man vorab schon angeben, um was es in dem jeweiligen Projekt gehen sollte, z.B. um Bildanalyse oder Anomalie Erkennung.

Im neu angelegten Projekt fängt man mit Datensammlung an und dafür muss die Datenquelle eingebunden werden. Neben einem Microcontroller könnte als Quelle auch ein Handy oder Cloudspeicher fungieren.

In diesem Kurs verwendete Beispiele benutzen ausschließlich Mikrocontroller als „Device“, wie es in Edge Impulse benannt wird.

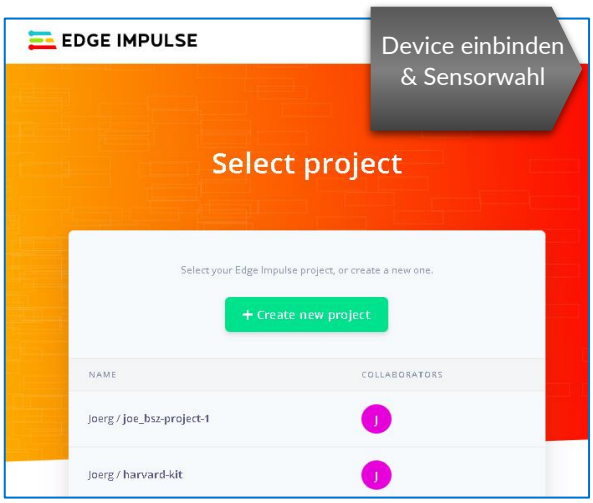


Abb. 37: Eigener Screenshot Edge-Impulse Webseite

Device in der Konsole anbinden:

Um die für das jeweilige Projekt benötigte Hardware einzubinden, muss Edge Impulse lokal aus der CMD-Konsole mit dem Befehl gestartet werden:

```
edge-impulse-daemon
```

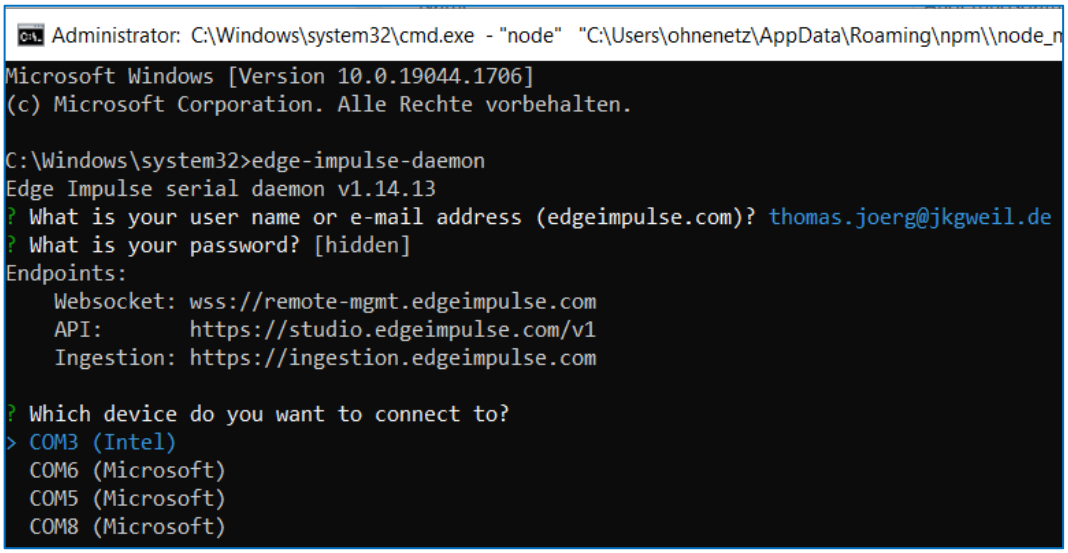


Abb. 38: Eigener Screenshot Kommandozeile

- Um die Verbindung mit einem Projekt aufzubauen, müssen Login-Daten eingeben werden
  Der COM-Port muss bestätigt werden - das Board (z.B. Arduino) wird mit dem Projekt verbunden.

Hinweis:
der Port kann entweder über die Arduino-Umgebung verifiziert werden

Oder über den Gerätemanager (Windows)

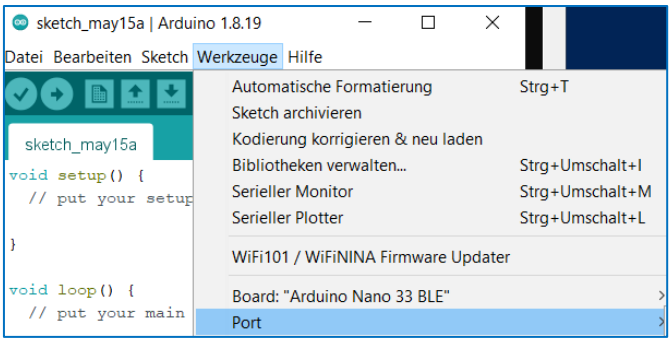


Abb. 39: Eigener Screenshot Arduino-IDE

Bei Fehlern bei der Verbindung ist ein Abbruch jederzeit möglich mit

```
Strg+C.
```

Nachdem die Fehler behoben sind, startet man die Einbindung erneut.

```
[SER] Failed to get info off device Timeout when waiting for >  (timeout: 5000) onConnected
[SER] Serial is connected, trying to read config...
[SER] Failed to get info off device Timeout when waiting for >  (timeout: 5000) onConnected

C:\Windows\system32>edge-impulse-daemon
Edge Impulse serial daemon v1.14.13
Endpoints:
    Websocket: wss://remote-mgmt.edgeimpulse.com
    API:       https://studio.edgeimpulse.com/v1
    Ingestion: https://ingestion.edgeimpulse.com

? Which device do you want to connect to? COM8 (Microsoft)
[SER] Connecting to COM8
[SER] Serial is connected, trying to read config...
[SER] Retrieved configuration
[SER] Device is running AT command version 1.7.0

Configuring API key in device... OK
[SER] 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? eumel
[WS ] Device "eumel" is now connected to project "JKGTeacher-project-1"
[WS ] Go to https://studio.edgeimpulse.com/studio/105535/acquisition/training to build your m
```

Abb. 40: Eigener Screenshot Kommandozeile

Es gilt: Zur gleichen Zeit kann nur ein Projekt pro PC aktiv sein.
Beim Wechseln des Projekts ist es notwendig, die Verbindungsdaten erneut einzugeben mit:

```
edge-impulse-daemon --clean
```

Wichtig: Das cmd-Fenster muss während der ganzen Arbeit mit Edge Impulse offenbleiben.

Hinweis: Die Verbindung zwischen dem Mikrocontroller und Edge Impulse ist empfindlich auf Wackelkontakte - die Verbindung wird ab und zu unterbrochen. Folgender Befehl ist dafür hilfreich:

```
edge-impulse-daemon port COM8
```

Device im Webbrowser anzeigen

Hat man sich auf der Webseite unter seinem Projekt eingeloggt, dann ist das Arduino-Board nun unter der Rubrik “Device” im Edge Impulse-Browserfenster sichtbar und aktiv.

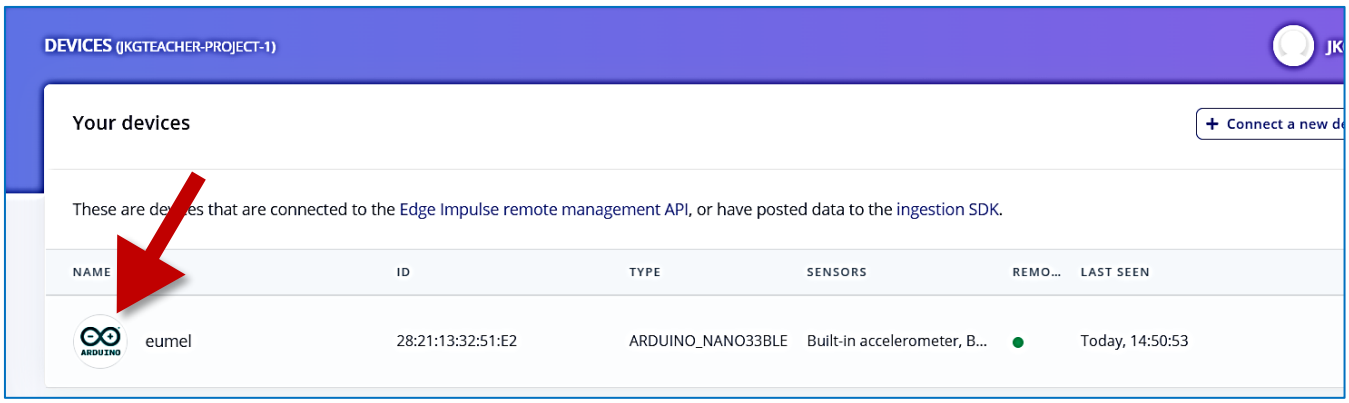


Abb. 41: Eigener Screenshot Edge-Impulse-IDE

Sensor auswählen

In diesem Dropdown-Menü stehen alle von Edge Impulse erkannte Sensoren zur Auswahl. Falls der ein- oder andere Sensor nicht angezeigt wird, hängt es von der jeweiligen Firmware ab.

ABER: Beim Arduino 33 BLE Sense sollte/darf das *eigentlich* nicht vorkommen, weil dieses Board so gut wie sonst kein anderes unterstützt wird (Stand Oktober 2022)

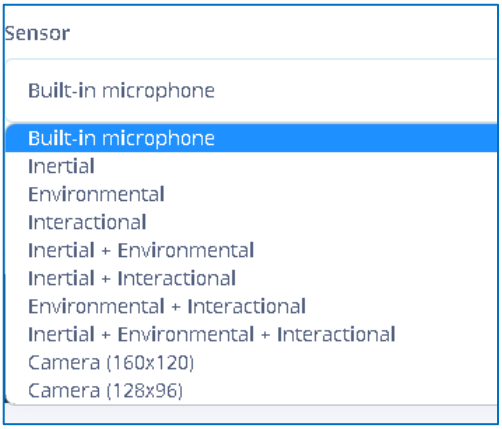


Abb. 42: Eigener Screenshot Edge-Impulse-IDE

Hier zwei Beispiele für Sensordaten:

Intertial

9 Achsen-Beschleunigungssensor (IMU)

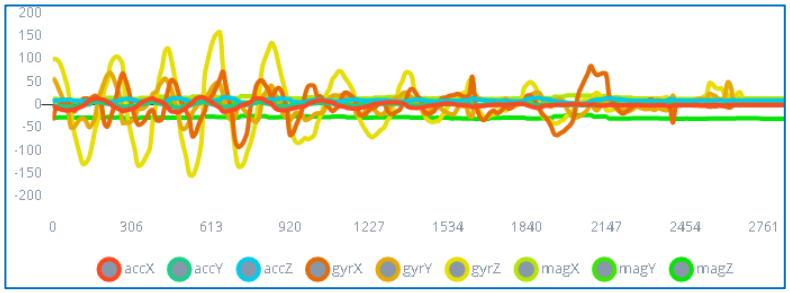


Abb. 43: Eigener Screenshot Edge-Impulse-IDE

Environmental + Interactional:

Zu den “Umgebungsgrößen” werden Temperatur, Feuchtigkeit und Luftdruck gezählt.

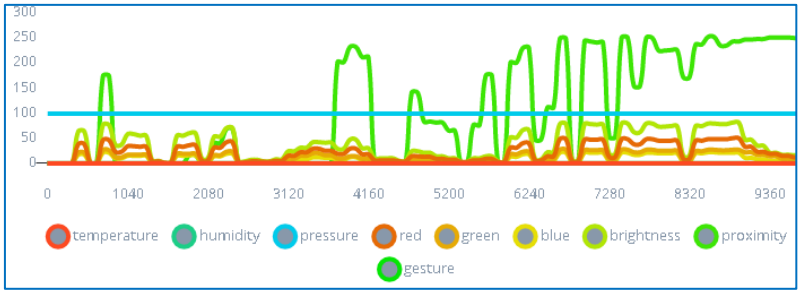


Abb. 44: Eigener Screenshot Edge-Impulse-IDE

Datensammlung (Data acquisition)

1. **Label festlegen**
   Wir wollen Trainingsdaten aufnehmen. Trainingsdaten zeichnen sich durch ‚Labels' aus, das heisst, sie sind nach ihren jeweiligen Klassen benannt. Nimmt man zum Beispiel Trainingsdaten von Früchten auf, dann werden Bilder von Bananen werden als ‚Banane' oder ähnlich gelabelt, Bilder mit Erdbeeren entsprechend usw.

2. **Parameter festlegen**
   Bei manchen Quellen, wie beim Audio- oder Beschleunigungssensor ist noch die Dauer eines Samplings (Beispiels) einzustellen. 10s als Standardeinstellung ist nicht immer passend. Es wird empfohlen, die Längere Sequenzen von z.B. mehreren gleichartigen Bewegungsmustern hintereinander als ein Sampling aufzuzeichnen und danach zu schneiden.

3. Mit "**`Start sampling`**" beginnt man die Aufzeichnung.

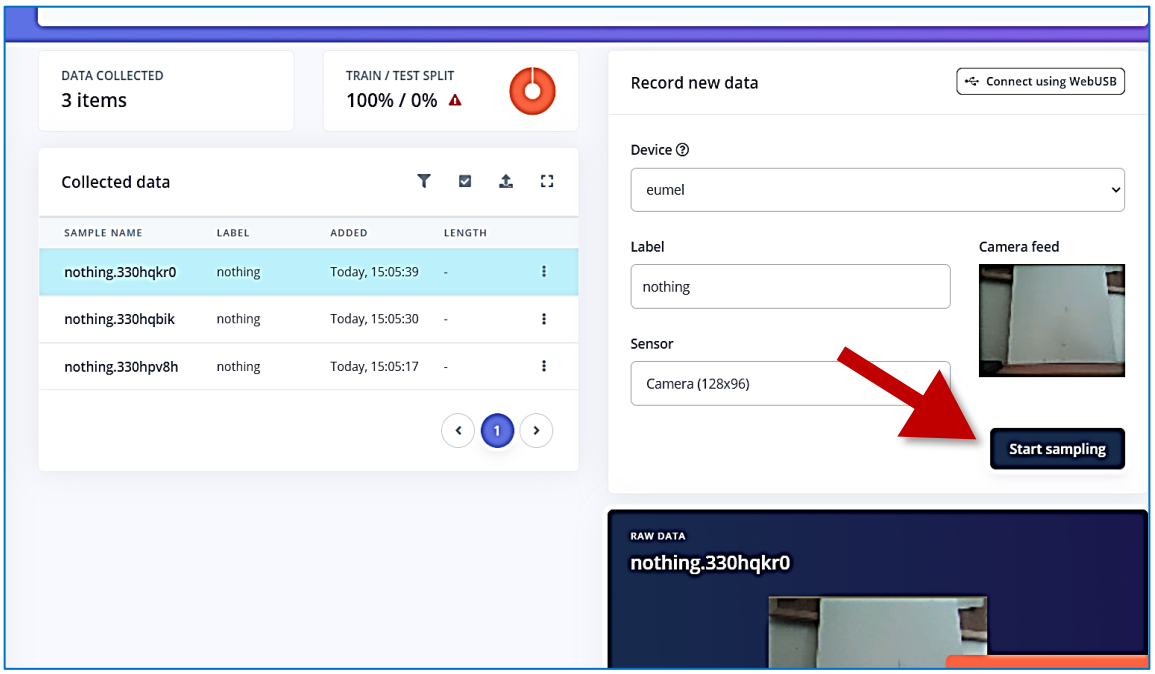


Abb. 45: Eigener Screenshot Edge-Impulse-IDE

Die aufgenommenen Daten müssen, soweit noch nicht bereits geschehen, in Train- und Test-Block aufgeteilt werden. Das geht auf drei verschiedene Art und Weisen:

a) einzeln aus dem Train- in den Testset verschieben (Move to test set)
b) über dem oberen Reiter in den Test-Bereich wechseln und zusätzliche Daten aufnehmen
c) automatisch vom Programm in 80% / 20% teilen lassen (über das (?) bei den Angaben zum Split)

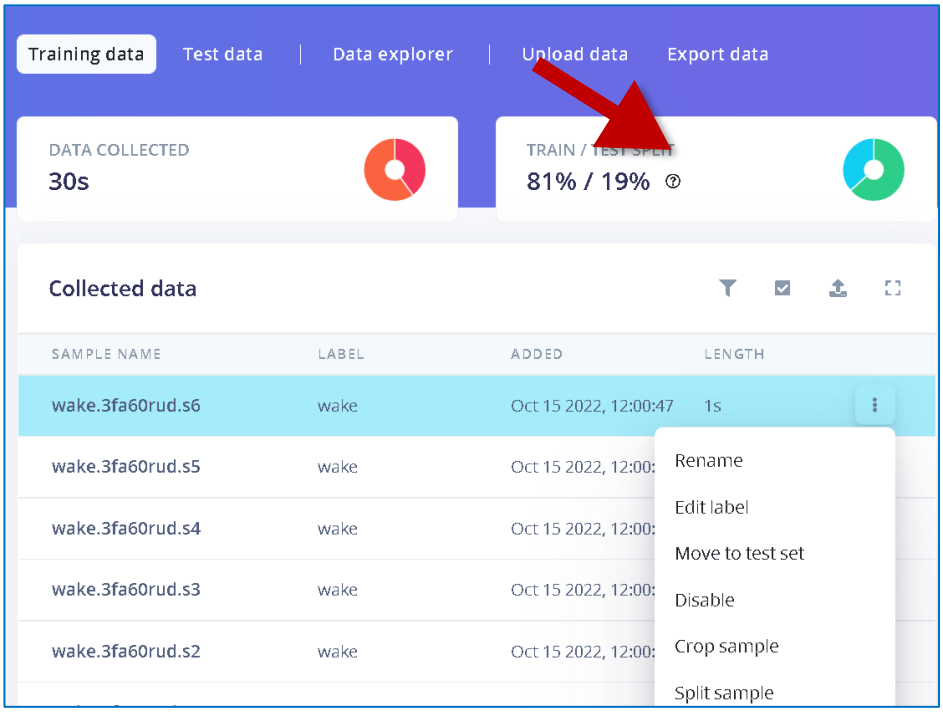


Abb. 46: Eigener Screenshot Edge-Impulse-IDE

Ebenfalls lassen sich in diesem Bereich die Daten exportieren oder die externen Daten hochladen.

Allgemeines zu Datensammlung

- **Abweichungen zwischen Training und Bereitstellung vermeiden**

Eine Abweichung zwischen Training und Bereitstellung tritt auf, wenn Sie die Trainingsdaten anders generieren als die Daten, die Sie zum Anfordern von Vorhersagen verwenden.

Die besten Ergebnisse erzielen Sie, wenn die Verteilung der Daten, mit denen Ihr Modell erstellt wurde, genau dem Unterschied zwischen dem Trainingsdataset und den Daten entspricht, für die Sie Vorhersagen in Ihrer Produktionsumgebung treffen.

- **Fehlende Werte vermeiden**

Wenn nur wenige Daten pro Klasse vorhanden sind, beeinträchtigt dies die Modellqualität.

Wenn die Daten zur Darstellung von Nullwerten Sonderzeichen verwenden, kann dies zu Problemen führen. Wenn Sie Daten aus einer CSV-Datei importieren, verwenden Sie leere Zeichenfolgen, um Nullwerte darzustellen.

- **Manuelle Aufteilung ist eventuell notwendig**

Die Daten für das Test-Dataset werden nach dem Zufallsprinzip ausgewählt. Bei unausgeglichenen Klassen kann es vorkommen, dass das Test-Dataset nur eine geringe Anzahl der Minderheitsklasse enthält, sodass das Training fehlschlägt.

Wenn Sie unausgeglichene Klassen haben, sollten Sie sie manuell aufteilen, damit in jeder Aufteilung genügend Zeilen mit den Minderheitsergebnissen enthalten sind.
Je mehr Features Sie zum Trainieren des Modells verwenden, desto mehr Daten müssen Sie bereitstellen. Für Klassifizierungsmodelle empfiehlt es sich, mindestens 10-mal so viele Datensätze wie Features zu verwenden.

Hinweise zu Audiodaten

Seien Sie sich bewusst, dass jede Art von Aufzeichnung (Datenschutz)rechtliche Folgen haben kann.

Sensoren, einschließlich Mikrofone, können in ihrer Fähigkeit, niedrigere und höhere Frequenzen zu erkennen sowie Geräusche aufzunehmen, sehr unterschiedlich sein. Daher kann die Wellenform desselben Geräuschs zwischen zwei verschiedenen Mikrofonen leicht unterschiedlich aussehen.

Beachten Sie, dass alle Samples im Datensatz exakt dieselbe Abtastrate, Bittiefe und Länge haben müssen.

Wenn Sie mit sogenannten „wake-words“ arbeiten, sollte der Datensatz aus drei Kategorien bestehen:

1. Hintergrundgeräuschen, die die Geräuschklasse bilden.
2. Die unbekannte Kategorie, d. h. alle Wörter, die nicht zu Ihrem Schlüsselwort oder Schlüsselwörtern gehören.
3. Schlüsselwörter. Dies sind die Wörter, auf die Ihr System reagieren soll.

Data Augmentation (Datenerweiterung)

- **Augmentierung von Audiodaten**
  Wenn man ein zufälliges Hintergrundgeräusch untermischt, hat man ein neues Sample erstellt. Diesen Vorgang kann man für verschiedene Hintergrundgeräusche wiederholen. Auf diese Weise vervielfacht man die Stichprobenmenge.

  Das ist zwar nicht so gut wie das Sammeln echter Proben in der tatsächlichen Zielumgebung, aber es kann hilfreich sein, wenn man nicht viele Daten zur Verfügung hat, mit denen man beginnen kann. Beim Audio kann man auch versuchen, an der Stelle, an der das Schlüsselwort erscheint, die Lautstärke oder die Tonhöhe zu ändern.

  In Edge Impulse stehen dafür folgende Funktionen zur Auswahl:

- Gaußsches Rauschen hinzufügen
- Maskierung zufälliger Frequenzbänder
- Maskierung zufälliger Zeitbänder
- Zufälliges Verzerren entlang der Zeitachse

- **Augmentierung von Bilddaten**
  Bei den Bilddaten erweitert man die Daten durch:

- Verzerrung,
- geänderten Blickwinkel,
- Farbkanäle,
- Größe des Bildes,
- Position des Objekts innerhalb des Bildes usw.

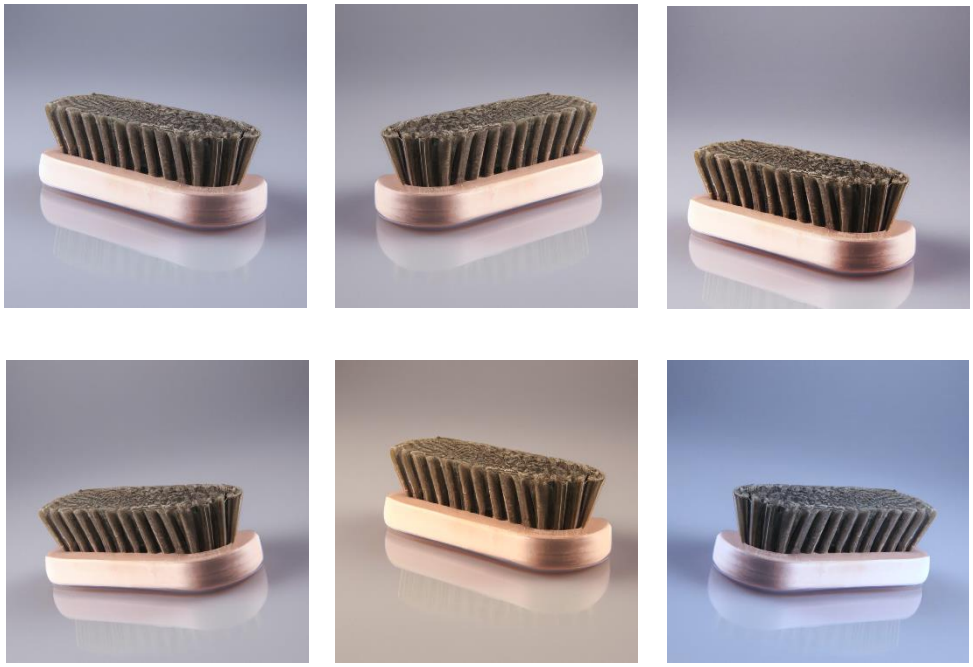
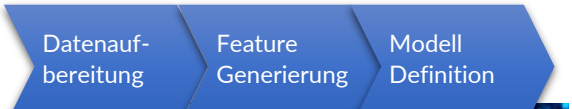


Abb. 47: Bild [Gemeinfrei] erzeugt mit DALL-E; Prompt „a shoebrush with studio background, professional food photo, high quality“ von Jörg [CC BY-SA 4.0 International]

Impulse design



Als Impulse Design bezeichnet man den Prozess von der initialen Verarbeitung der aufgenommenen Daten bis zur Erstellung des Neuronalen Netzes & Hyperparameter-Einstellung.

Bevor das Neuronale Netz mit den Daten gefüttert wird, müssen zunächst die Features generiert und die Parameter definiert werden.

Das Neuronale Netz wird von Edge Impulse mit Standardwerten vorkonfiguriert, mit denen man zunächst starten kann. Die Netztopologie wird man aber sehr wahrscheinlich verändern und anpassen müssen.

In der linken Menüleiste unter “Impulse Design” werden einzelne Bausteine der Datenaufbereitung und Algorithmen zusammengestellt und gespeichert.

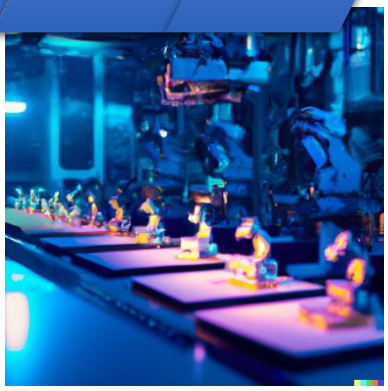


Abb. 48: Bild [Gemeinfrei] erzeugt mit DALL-E; Prompt „a assembly line in a factory produces tiny robots, professional studio photo, vivid lighting“ von Jörg [CC BY-SA 4.0 International]

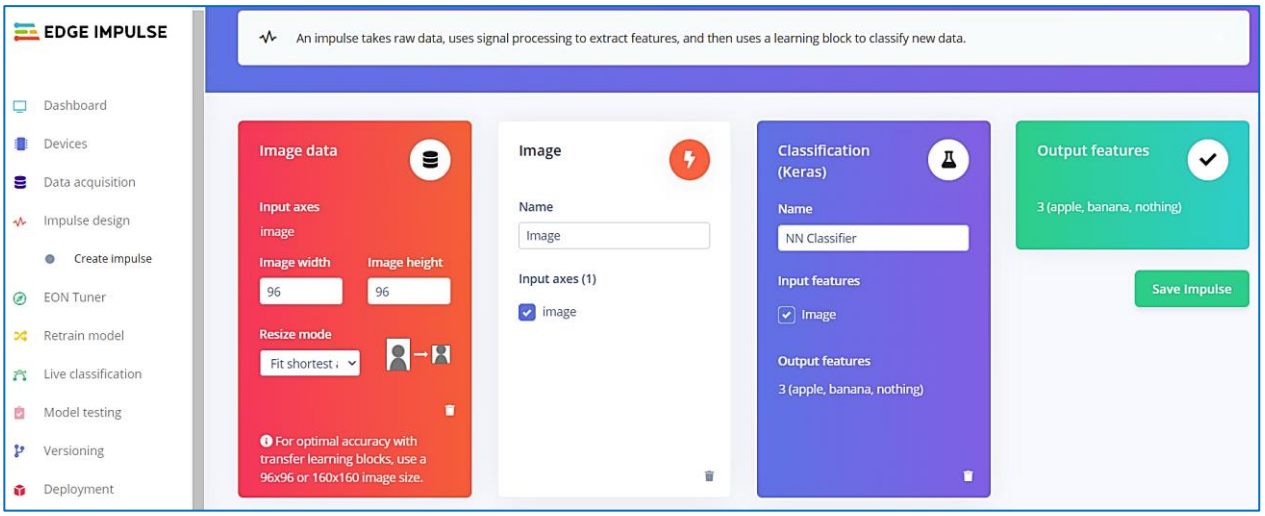


Abb. 49: Eigener Screenshot Edge-Impulse-IDE

Die Übersicht an Blocks und weiterführende Informationen zu den jeweiligen Blocks findet man hier: https://docs.edgeimpulse.com/docs/edge-impulse-studio/impulse-design

Datenaufbereitung/Parameter-Einstellung

Entsprechend der Art der Rohdaten müssen diese zuerst parametrisiert bzw. formatiert oder konvertiert werden. Aufgenommene Bilder müssen z.B. skaliert werden, bei Audiosignalen werden Frequenz und Aufzeichnungsdauer festgelegt. Diese Einstellungen nennt Edge Impulse „Parameter“.

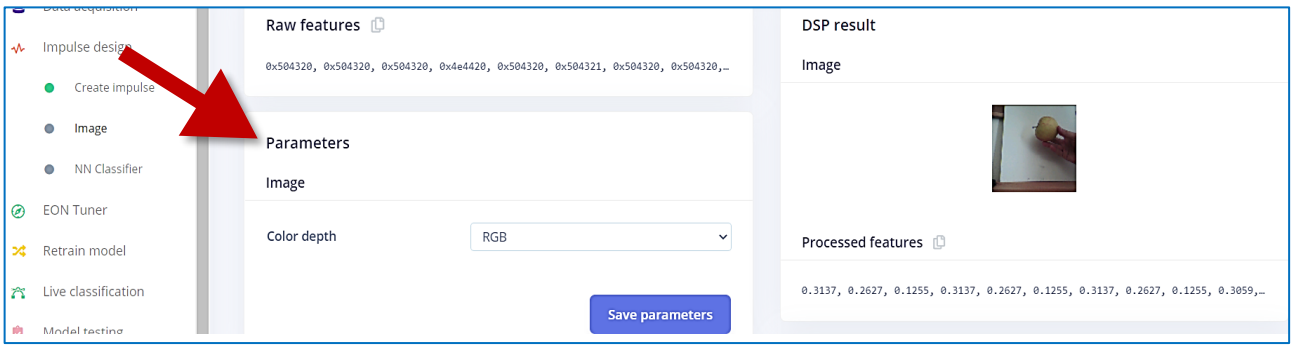


Abb. 50: Eigener Screenshot Edge-Impulse-IDE

Feature-Generierung

Nachdem die Parameter gespeichert worden sind, können Features generiert werden. Die generierten Features werden im „Feature Explorer“ angezeigt, wobei man eventuell schon die Trennung der Klassen erkennen könnte.

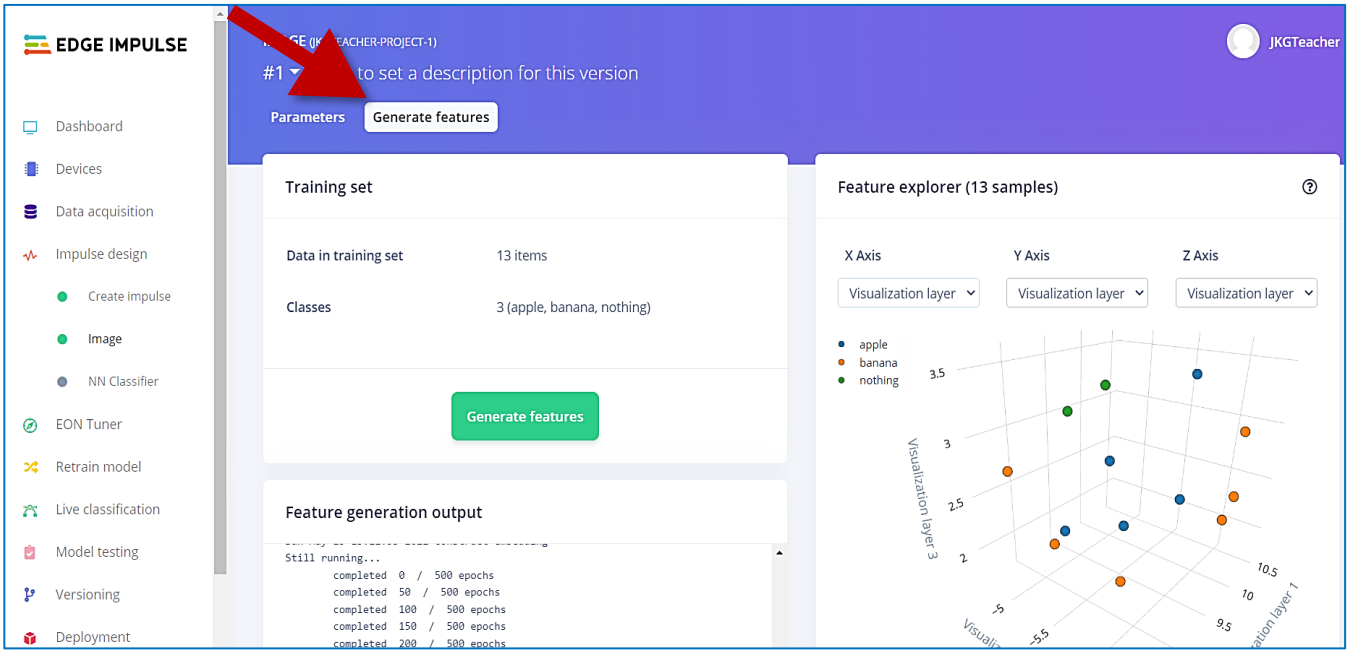


Abb. 51: Eigener Screenshot Edge-Impulse-IDE

Was sind „Features“ (Merkmale/Attribute)?

Aus den Rohdaten müssen geeignete Merkmale ausgewählt werden, um das ML-Modell so klein und schnell wie möglich zu halten. Da die manuelle Auswahl der Merkmale bei komplexen Datensätzen nicht möglich ist, greift man auf fertige Funktionen zurück.

1. Statt einige der Rohdaten auszuwählen, kann man die Proben auf verschiedene Weise auswerten, um einzigartige Merkmale zu erzeugen, die helfen zu beschreiben, was im System passiert. Zum Beispiel kann man den quadratischen Mittelwert (RMS) für alle Stichproben in jeder Achse berechnen. Dies ist eine einfache Berechnung, die eine einzelne Zahl pro Achse und so etwas wie einen Durchschnitt oder Mittelwert für alle diese Zahlen in jeder Achse liefert.

2. Eine gängige Technik bei der Betrachtung von Vibrations- oder Bewegungsdaten ist die Fourier-Transformation dieser Daten, um Informationen über sie im Frequenzbereich zu erhalten. Die FT ist eine Möglichkeit, ein Signal in seine verschiedenen Frequenzkomponenten zu zerlegen.

3. Die spektrale Leistungsdichte (PSD) ist ein weiterer guter Satz von Merkmalen, vor allem, wenn es um Bewegungs- und Vibrationsdaten geht.

NN-Classifier: Netztopologie des Neuronalen Netzes definieren

Als nächstes kommen die Learning-Blocks. Hier wird z.B. Neuronales Netz konfiguriert.

Die Möglichkeiten sind mannigfaltig und bilden den Workflow von Tensorflow oder Pytorch ab. Eine genauere Beschreibung kann aufgrund des umfangreichen theoretischen Überbaus hier nicht gegeben werden.

Glücklicherweise bietet Edge Impulse einen Assistenten an, der bei der Erstellung der initialen Netztopologie unterstützt. Von hier aus können die Hyperparameter dann weiter verfeinert und angepasst werden.

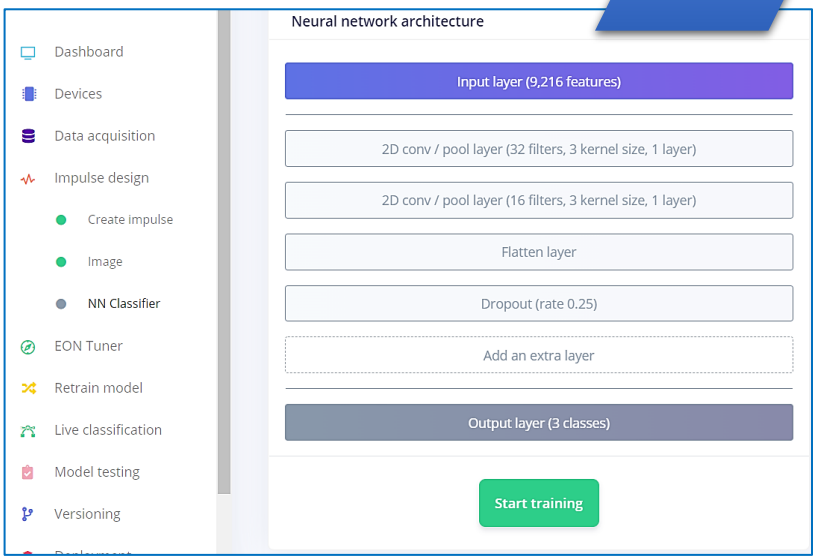


Abb. 52: Eigener Screenshot Edge-Impulse-IDE

Training

Das Training verläuft in mehreren Phasen, in die man aber nicht mehr eingreifen kann. Alles verläuft online, also in der Cloud auf den Edge Impulse Servern. Deshalb kann es manchmal etwas dauern, bis das eigene Projekt in der Queue vorne liegt und gestartet werden kann.

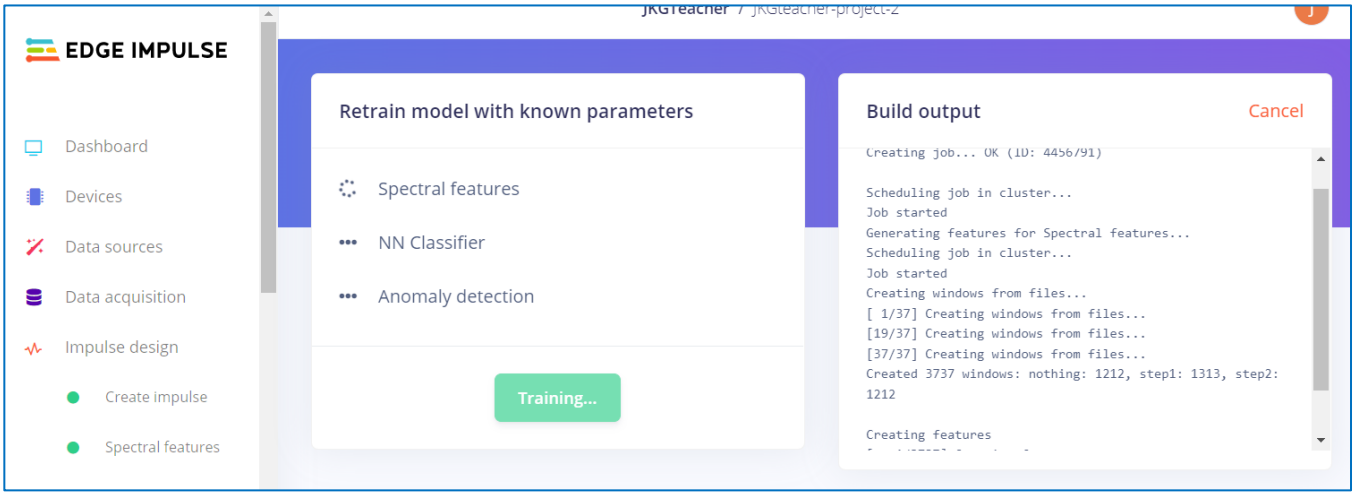


Abb. 53: Eigener Screenshot Edge-Impulse-IDE

Live Inference: Modell in Echtzeit ausprobieren

Wenn man mit den Ergebnissen des Trainings zufrieden ist, kann man das Model testen, und zwar auf unterschiedliche Art und Weise:

Abb. 54: Bild [Gemeinfrei] erzeugt mit DALL-E; Prompt „a pug playing nintendo switch, steampunk version, digital art“ von Jörg [CC BY-SA 4.0 International]

Über eine GUI im Bereich „Live Classification“:

1) Mit den Daten aus dem Test-Datensatz (Classify existing test sample) über „Load sample“

Abb. 55: Eigener Screenshot Edge-Impulse-IDE

2) Mit Live-Daten über „Start sampling“

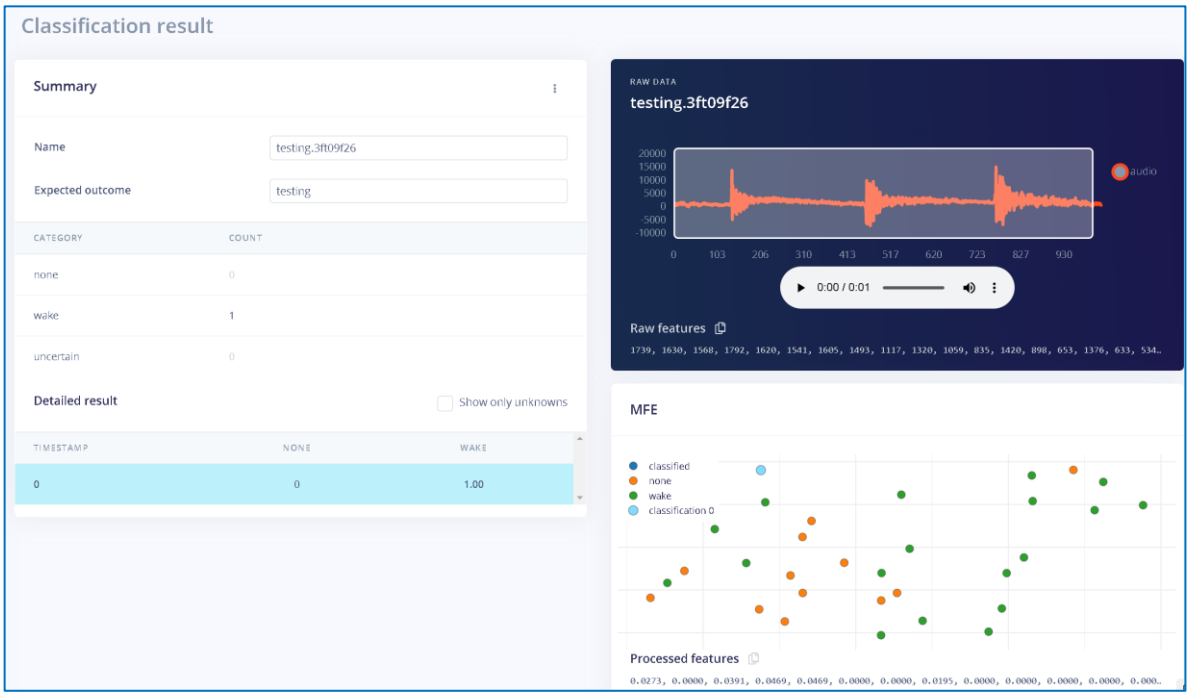


Abb. 56: Eigener Screenshot Edge-Impulse-IDE

Über die Konsole:

Tests startet man mit dem folgenden Befehl:

```
edge-impulse-run-impulse
```

Hinweis: Zu diesem Zeitpunkt muss der Mikrocontroller noch mit Edge Impulse-Firmware bespielt, aber nicht mehr aktiv sein.

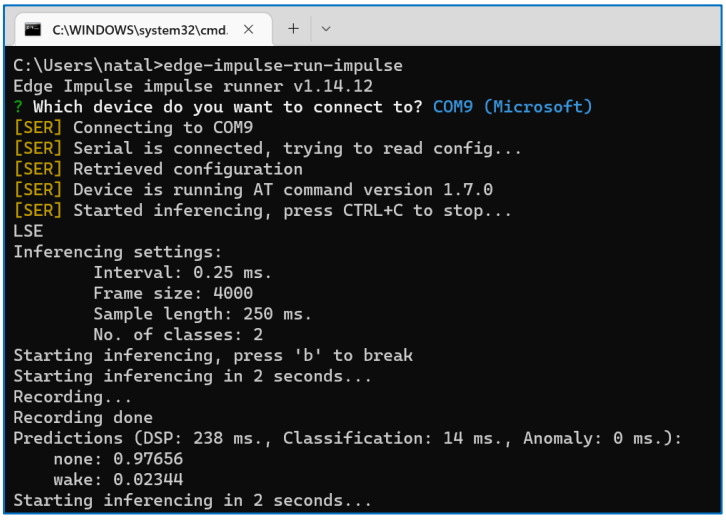


Abb. 57: Eigener Screenshot Kommandozeile

EON Tuner: ”KI unterstützt KI”

Wenn die Genauigkeit des Models (Accuracy) nicht überragend seine sollte, hilft „EON Tuner“, das beste Modell für Ihre Anwendung zu finden, das den Einschränkungen des konkreten Mikrocontrollers entspricht. Es werden hierbei auch Vorschläge zu den Prozess-Blöcken gemacht - z.B. Spektralanalyse dazu zu nehmen.

Manuelles Testen des erzeugten Modells ist weiter unten in der Navigation unter dem Punkt “Live Classification" möglich

Abb. 58: Bild [Gemeinfrei] erzeugt mit DALL-E; Prompt „a cute little robot thinking himself, digital art, high quality 3d render“ von Jörg [CC BY-SA 4.0 International]

Wiederholung des Zyklus

Sehr wahrscheinlich wird man im ersten Anlauf kein vernünftig arbeitendes Modell erstellen können. Deshalb wird man die oben beschriebene Pipeline immer wieder überarbeiten müssen,

- Durch mehr Testdaten oder
- Durch Ausbalancierung der Zusammensetzung der Datensätze,
- Durch unterschiedliche Sampling-Raten,
- Durch veränderte Feature-Generierung
- Durch angepasstere Topologie des Neuronalen Netzes
- Durch unterschiedliche Parameter des K-Means (Anomaly Detection)
- Durch Hyperparameter-Tuning.
- Usw.

Abb. 59: Bild [Gemeinfrei] erzeugt mit DALL-E; Prompt „a hamster running in a hamsterwheel, cyberpunk version, digital art“ von Jörg [CC BY-SA 4.0 International]

Hier können keine substantiellen Tipps gegeben werden, weil hier vieles auf Erfahrung und Ausprobieren basiert. Edge-Impulse ermöglicht durch den sehr schlanken Ablauf der Pipeline das Experimentieren mit den einzelnen Prozess-Schritten. Der Schüler sollte auch genau das nutzen.

Modell-Deployment (Veröffentlichen/Exportieren)

Für unterschiedliche Boards stehen hier unterschiedliche Lösungen zur Verfügung. Die am leichtesten nutzbare Version findet sich auch hier wieder für den Arduino 33 BLE Sense: Der von Edge Impulse erzeugte Arduino-Code kann direkt in die Arduino-IDE eingebunden und kompiliert werden:

Variante A) Modell als Arduino-Bibliothek bereitstellen

Wählen Sie im Menüpunkt „Deployment“ bei den Bibliotheken “Arduino” und scrollen Sie zum unteren Ende der Seite, um mit „Build“ die Bereitstellung zu starten.

Schritt 1) Herunterladen der Bibliothek

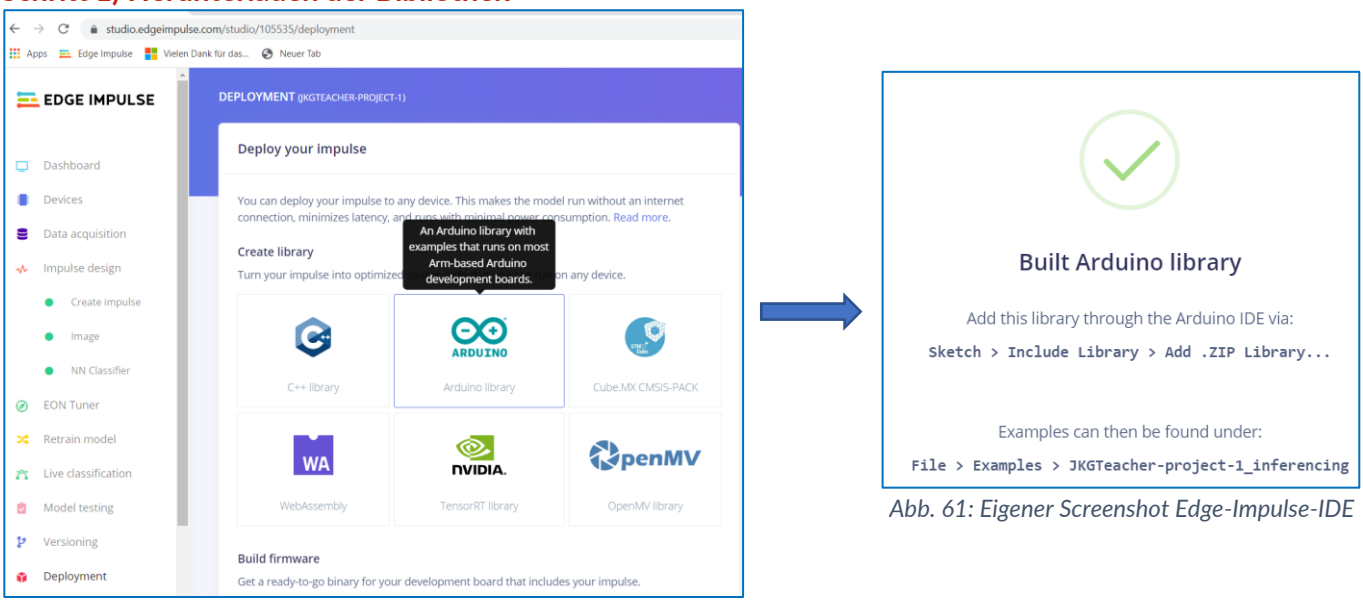


Abb. 60: Eigener Screenshot Edge-Impulse-IDE

Abb. 61: Eigener Screenshot Edge-Impulse-IDE

Schritt 2) Einbinden in die IDE

https://docs.edgeimpulse.com/docs/deployment/running-your-impulse-arduino

Die beim Deployment erstellte Bibliothek, die den Namen des Projekts trägt, muss in die Arduino-Umgebung als .zip-Bibliothek hinzugefügt werden:

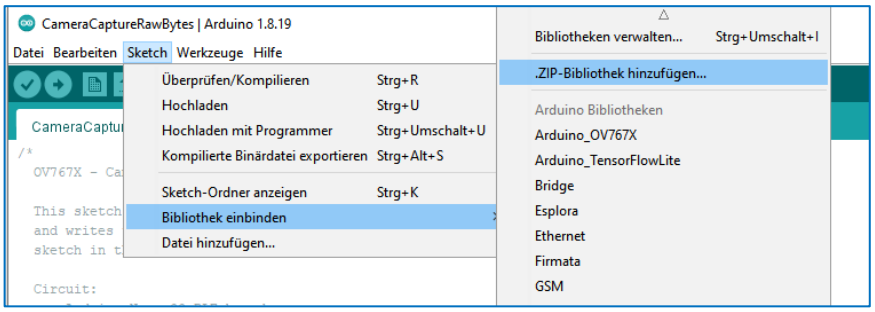
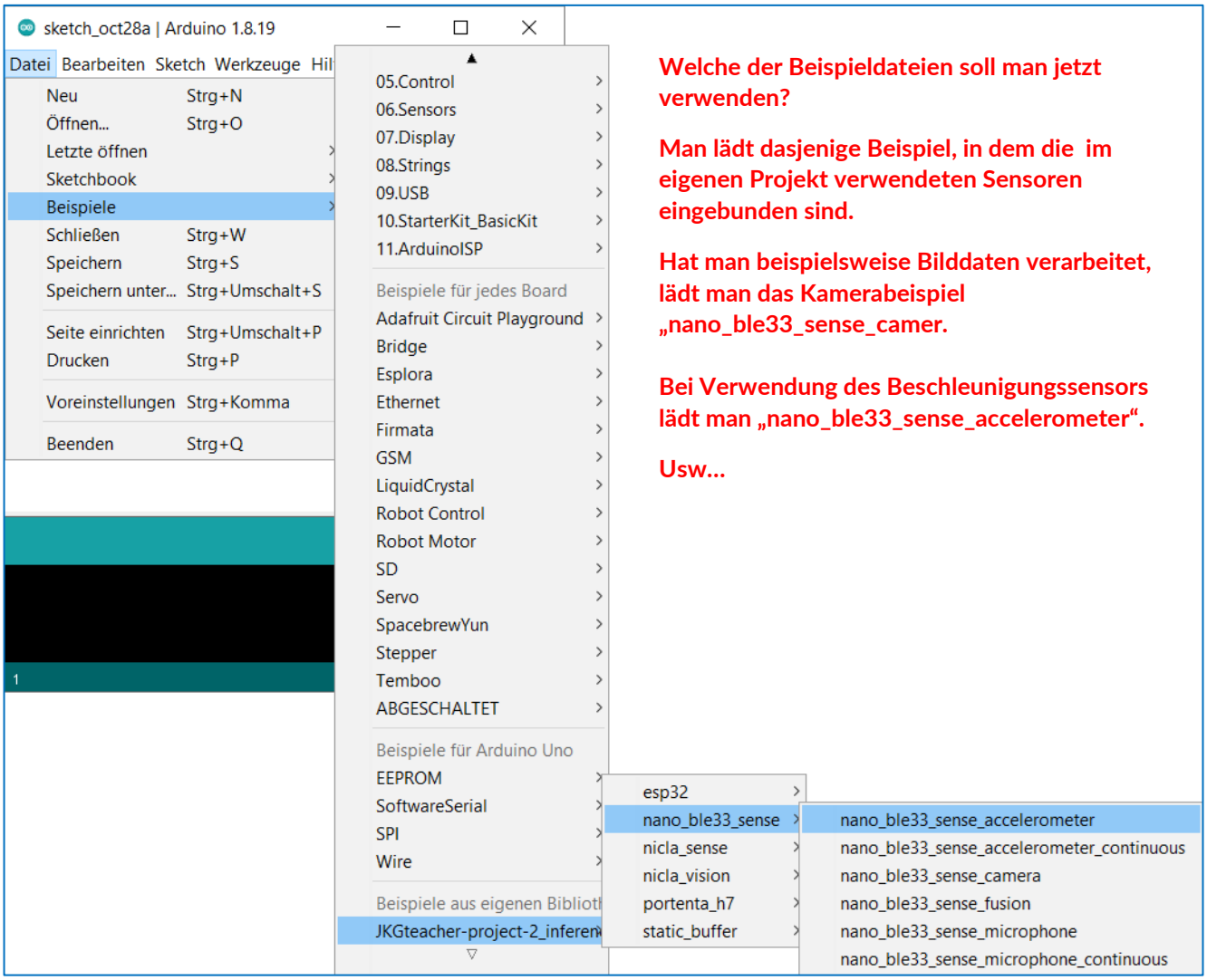


Abb. 62: Eigener Screenshot Edge-Impulse-IDE

Die mitinstallierten Beispiele sind ohne weiteres lauffähig. Nachdem man die zum Model passende „.ino“-Datei auf das Arduinoboard hochgeladen hat, ist der Mikrocontroller als Device nicht mehr für Edge Impulse erreichbar, denn dann ist die Edge-Impulse-Firmware überschrieben.



Welche der Beispieldateien soll man jetzt verwenden?

Man lädt dasjenige Beispiel, in dem die im eigenen Projekt verwendeten Sensoren eingebunden sind.

Hat man beispielsweise Bilddaten verarbeitet, lädt man das Kamerabeispiel „nano_ble33_sense_camer.

Bei Verwendung des Beschleunigungssensors lädt man „nano_ble33_sense_accelerometer“.

Usw...

Abb. 63: Eigener Screenshot Arduino-IDE

Kurzer Auszug aus dem Beispielquelltext:

```
#include <JKGteacher-project-2_inferencing.h>

#include <Arduino_LSM9DS1.h>

#define CONVERT_G_TO_MS2    9.80665f
#define MAX_ACCEPTED_RANGE  2.0f
static bool debug_nn = false;

void setup()

{   Serial.begin(115200);
    Serial.println("Edge Impulse Demo");
}

void loop()

{    ei_printf("\nStarting inferencing in 2 seconds...\n");
```

In dieser header-Datei, die den selben Namen trägt wie das Edge-Impulse-Projekt, sind alle weiteren Querverweise auf Modelle, wie zum Beispiel CNNs, Neuronale Netze, k-Means-Cluster oder FFTs usw.

Wie hier zu sehen ist, entspricht der Aufbau des Quelltextes den üblichen Arduino-Gepflogenheiten. Hier kann man eigene Änderungen vornehmen.

Variante B) Firmware generieren

Alternativ kann auch direkt das entsprechende Firmware generiert werden. Dabei wird ein Archiv heruntergeladen, wo neben der bin-Datei auch ein Bootloader einhalten ist.

1) Firmware herunterladen

Abb. 64: Eigener Screenshot Edge-Impulse-IDE

2) Firmware mit Arduino-CLI aufspielen

Um die neu erstellte Firmware auf das Board zu laden, kopieren Sie bitte in den entpackten Ordner die Datei arduino_cli.exe. Umgekehrt ist es nicht empfehlenswert, die neue Firmware in den arduino-cli-Ordner zu kopieren. Dort bereits enthaltene Edge Impulse-Firmware wird immer wieder benötigt, um neue Projekte zu erstellen oder das angefangene Projekt zu bearbeiten.

Modell anpassen

Abb. 65: Bild [Gemeinfrei] erzeugt mit DALL-E; Prompt „a cute little robot cleaning the floor, white background, digital art“ von Jörg [CC BY-SA 4.0 International]

Wenn neue Daten gesammelt werden müssen, muss zuerst die Firmware wieder auf den Mikrocontroller aufgespielt werden.

Wenn nach den vorgenommenen Änderungen eine neue Version der Bibliothek exportiert wird, ist folgendes zu beachten:

- Zuerst muss die alte Bibliothek manuell aus dem Ordner *C:\Users\XYZ\Documents\Arduino\libraries* entfernt werden. Überschreiben der vorhandenen Bibliothek ist nicht möglich.
- Erst dann kann die neue Version der Bibliothek aus der .zip-Datei installiert werden
- Falls die Beispielprogramme benutzt werden, ist darauf zu achten, dass die Beispiele immer aus der aktuellen Bibliothek stammen. Da die Bibliothek immer den Namen des Projekts trägt, ist die Verwechselungsgefahr groß.

AddOn/Ausblick: Edge-Impulse: Object Detection auf ESP-Eye & Arduino Serial Monitor

Abb. 66: Eigener Screenshots aus Edge-Impulse IDE

Auf dem ESP-Eye kann man auf Edge-Impulse selbst trainierte Object-Detection-Modelle erstellen. Diese lassen sich nach dem Download der Projektdateien sogar in der Arduino-Entwicklungsumgebung kompilieren! Das bedeutet, dass man von hier aus die Skripten selbst weiterentwickeln kann.

Um Object-Detection nutzen zu können, muss zunächst in der Projekteinstellung links unten unter ‚Project Info' das Dropdown auf „Bounding Boxes" gestellt werden.

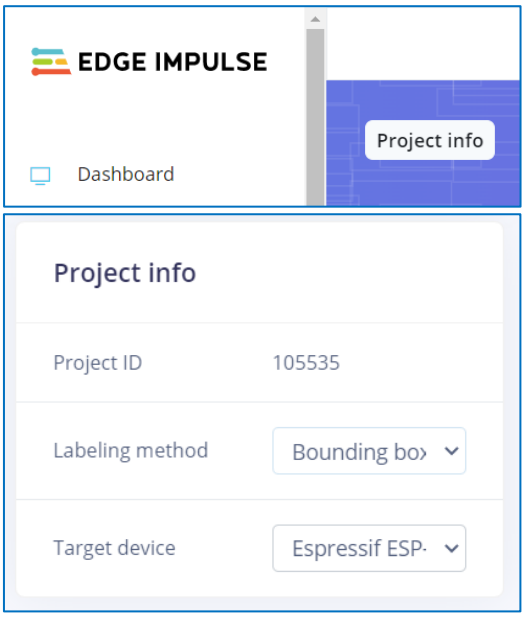


Abb. 67: Eigener Screenshots aus Edge-Impulse IDE

Anschließend nimmt man eine Reihe von Bildern auf, die unmittelbar im Anschluss selbstständig händisch mit Rahmen versehen werden. Da das viel Arbeit bedeutet, wäre es vielleicht eine Überlegung wert: Man erstellt die Bilder eventuell synthetisch – also zum Beispiel per Skript.

Das Setup für das Testprojekt könnte zum Beispiel so aussehen:

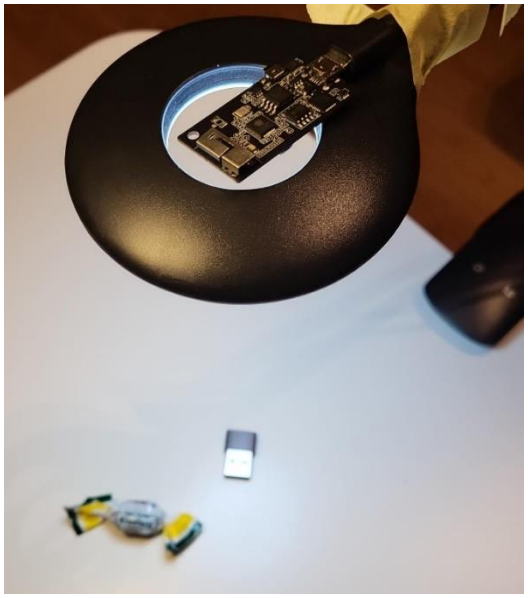


Abb. 68: Eigenes Foto vom Setup

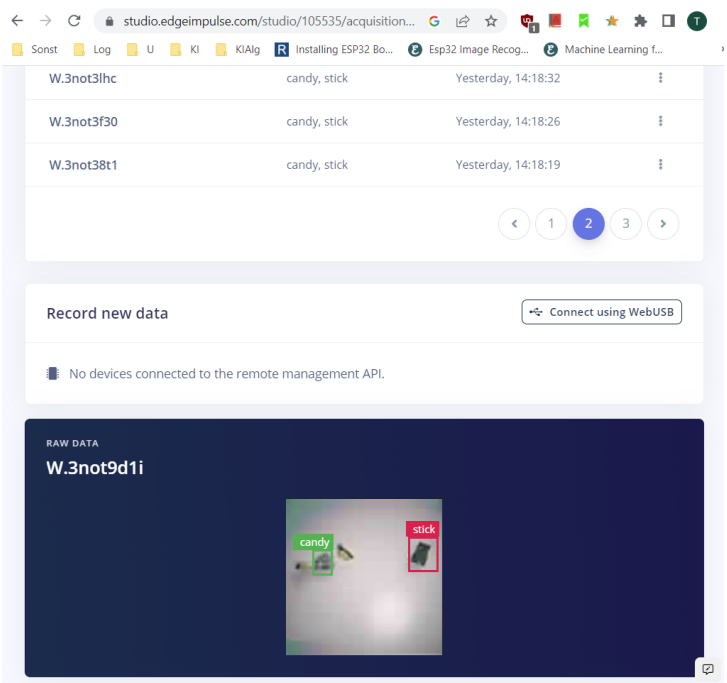


Abb.69: Eigener Screenshots aus Edge-Impulse

Deployment

Lädt man anschließend unter ‚Deployment' die Arduino-Dateien herunter, so kann man das ESP32-Projekt direkt laden und kompilieren (vorausgesetzt, man hat die entsprechenden Board-Informationen zum ESP32 heruntergeladen und installiert. Falls nicht, hier nochmals der Downloadlink, den man in die ‚Voreinstellungen' der Arduino-IDE einfügen muss:

https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json

Anschließend kann man die ‚korrekten' Board-Daten laden. Stand März 2023: Version 2.07 (!)

In der Arduino-IDE zeigt sich nach der Kompilierung das folgende Bild:

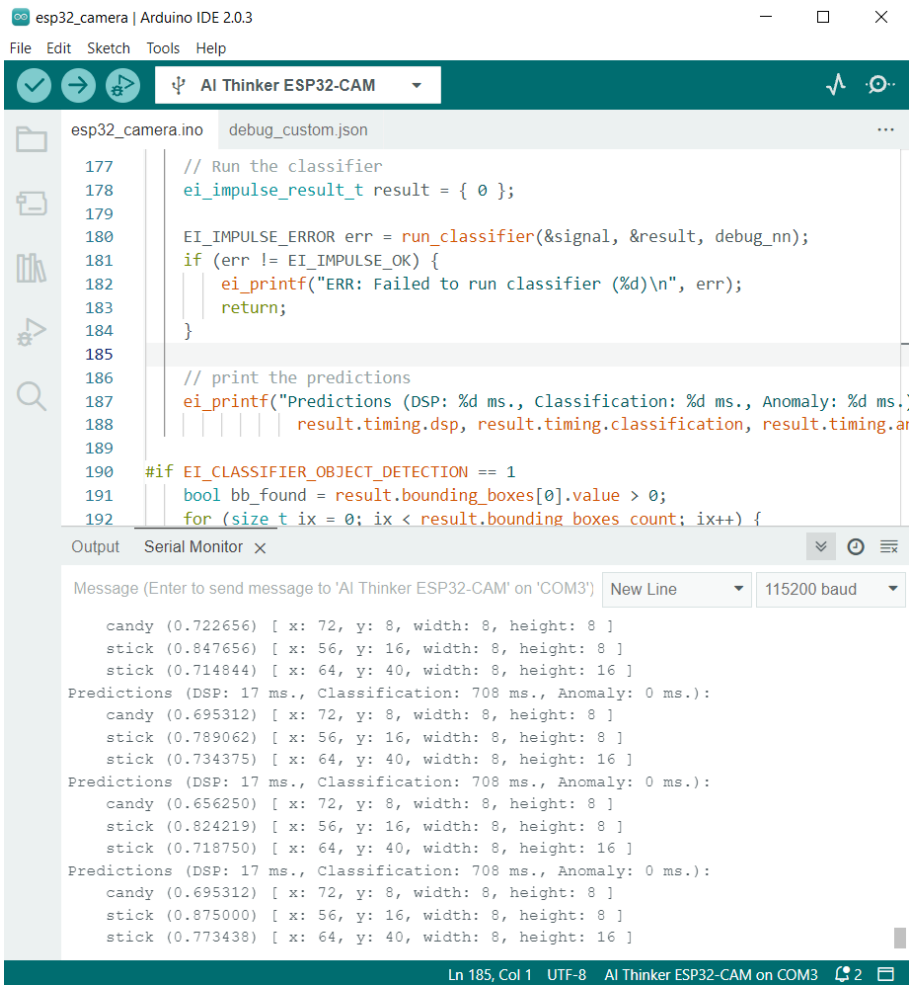


Abb. 70: Eigener Screenshot Arduino-IDE