Projekt:

Intelligent Wildlife Watch

von

Lucy Fischer

Betreuungslehrer: Herr Mertens, Herr Nebe

Städt. St. Michael Gymnasium, Bad Münstereifel

Inhaltsverzeichnis

Abbildungsverzeichnis

1. Einleitung

1.1. Kurzfassung

Künstliche Intelligenz und Natur kann das zusammenpassen? Die „Intelligent Wildlife Watch“ macht es möglich.

Sie nutzt eine KI, um Tiere zu schützen und ungewollte Kollisionen zwischen Wildtieren und Menschen zu verringern. Dank Objekterkennung können Tiere gesichtet, kategorisiert und lokalisiert werden. Dadurch werden nicht nur der Umweltschutz und die differenzierte Beobachtung der faunistischen Biodiversität unterstützt, sondern auch die Arbeit von Forstämtern und Tierschützern wird vereinfacht. Durch die Nutzung von Infrarotstrahlung (IR) können auch nachtaktive Tiere aufgenommen werden, was zu noch präziseren Beobachtungsergebnissen führt. Weiter kann durch die Nutzung von IR eine „Rund um die Uhr-Sichtung“ sichergestellt werden.

Vorerst liegt der Hauptfokus auf Wölfen, Rehen und Wildschweinen.

1.2. Ideenfindung

Der erste Gedanke war es ein Nachtsichtfernglas, welches aktuell tausende von Euros kostet, günstiger herzustellen. Jedoch ergab sich nach Vorversuchen und Nachforschungen, dass dies aufgrund der Spezialen Linsen kaum möglich ist. Daher suchte ich nach einer neuen Projektidee und stieß dabei auf einen Bericht über die Gefährdungen von Wölfen. Daraufhin überlegte ich, ob es eine Möglichkeit gäbe ihnen zu helfen und ja die gibt es. Für diese Projektidee war es sogar möglich den Nachtsichtaspekt von der vorherigen Idee sinnvoll einzusetzen und dafür die bereits erworbene ESP32CAM und ihre besondere Linse zu nutzen.

Nachdem ich den ersten Bericht über Wölfe gelesen hatte, fiel mir auf, dass heutzutage immer mehr Artikel mit einem Bezug zu Wölfen erscheinen. Einerseits geht es darum, wie gefährdet sie sind und anderseits, dass sie Probleme bereiten.

Laut GEO (GEO, 2021) hat sich die Anzahl der überfahrenen Wölfe in Deutschland mehr als verdreifacht. Durch frühzeitige Sichtung der Wölfe wäre es möglich Autofahrer zu warnen und somit nicht nur die Wölfe, sondern auch die Autofahrer zu schützen.

Zudem ist die Anzahl von unrechtmäßig getöteten Wölfe gestiegen. Auch in diesem Fall ist es möglich sie besser zu schützen, wenn man weiß, dass es sie gibt und wo sie ungefähr sind.

Jedoch gibt es auch viele Fälle von gerissenen Nutztieren durch Wölfe. Um präventive Maßnahmen einleiten zu können, wäre eine frühzeitige Sichtung, welches durch die „Intelligent Wildlife Watch“ sichergestellt werden kann, hilfreich.

Auch würde eine automatische Erkennung es für Förster und Umweltschützer leichter machen die Wölfe zu zählen und den Überblick zu behalten, wo sie sind. Auch Jäger könnten davon profitieren, in dem sie z.B. bestimmte Gebiete bei der Jagd meiden. Auch Spaziergänger könnten besser gewarnt werden, so dass ihnen nichts passiert und sie die Wölfe nicht stören.

Auf diese Überlegungen folgte der Gedanke, dass die Erkennung von Wölfen auch auf andere Tiere erweitert werden könnte. Dies würde Förstern, Jägern und Umweltschützern ein noch breiteres Anwendungsspektrum bieten, denn dadurch wird es leichter werden den jeweiligen Tierbestand zu zählen und zu kontrollieren, sowie Tiere ausfindig zu machen.

Weitere Tiere, wie z.B. Wildschweine, verursachen erhebliche Schäden und auch hier wäre eine rechtzeitige Sichtung von Vorteil.

Daher habe ich mich entschlossen den Fokus zunächst auf drei Wildtiere den Wolf, das Reh und das Wildschwein zu legen.

2. Umsetzung

2.1. Vorgehensweise

Nachdem sich die oben beschriebene Idee geformt hatte, ging es an die Umsetzung. Dafür musste ich mir überlegen, wie vorzugehen ist und welche Dinge ich benötige. Ich musste viele Informationen zur Objekterkennung und KIs sammeln, sowie Grundlagen der Programmierung erlernen.

Basierend auf meinem neuen Wissen, suchte ich eine passende IDE (Integrated Development Environment), einen Mikrokontroller und eine gute Kamera. Damit eine KI Objekterkennung lernen kann braucht sie hunderte von Bildern. Deswegen musste ich mir überlegen, ob ich die benötigten Bilder aus dem Internet nehme oder selber mache und falls ich diese selber mache, dann wie. Da es nicht möglich ist die Objekterkennung von echten Tieren zu demonstrieren, musste eine andere Lösung her. Meine Lösung letztendlich ist es für den Prototypen Spielzeugtiere zu verwenden. Nachdem ich diese gefunden hatte, war der nächste Schritt für die Kamera meiner Wahl, die ESP32CAM, den benötigten Programmcode zu schreiben, damit sie genutzt werden kann.

Der darauffolgende Schritt war es Bild-Versuche durchzuführen und die Settings bzw. den Programmcode entsprechend der Ergebnisse anzupassen.

Schon bei meinen ersten Nachforschungen zur Objekterkennung, war mir Edge Impulse (siehe „Edge Impulse“) aufgefallen. Also lernte ich, wie es verwendet wird und lud meine aufgenommenen Bilder hoch. Diese benannte bzw. beschriftete ich, was die Grundlage für meine KI bildete. Als nächstes musste die KI trainiert und implementiert werden. Anschließend mussten Versuche durchgeführt und Probleme behoben werden.

Zusätzlich war es wichtig eine Hülle für die ESP32CAM herzustellen, da zuvor schon eine Kamera kaputt gegangen war, weil ein adäquater Schutz fehlte. Damit die Hülle allen Anforderungen entspricht und dazu noch die richtigen Maße hat muss ich sie selbst erstellen. Dies mache ich mit dem Programm FreeCAD und drucke es dann mit einem 3D-Drucker aus.

Ein zusätzlicher Schritt war es zukünftige Pläne und Ideen für die Verbesserung des Projektes zu erstellen.

2.2. ESP32CAM

Die ESP32CAM besteht aus dem OV2640 Kameramodul, sowie dem ESP32 Wifi-Mikrokontroller. Sie nutzt eine Versorgungspannung von 5V bei einer Frequenz von 160MHz. Für das Projekt wurde eine besondere Linse verwendet. Diese ist nicht nur weitwinklig, sondern auch für den Infrarot-Bereich ausgelegt. Für die Nachtsicht wird eine externe Infrarotquelle verwendet. Dafür nutzen wir das IR-LED-Platinen Modul von POFET.

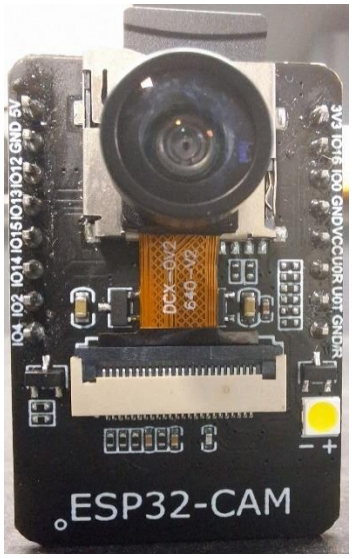

Abbildung 1: ESP32-CAM

Abbildung 2:IR-LED-Platinen Modul *(Amazon)*

Im Vergleich zu anderen Modellen hat sie einige Vorteile:

- Integriertes Bluetooth
- Integriertes Wlan
- Ist kompakt und somit leichter zu transportieren. Das macht es für die Anwendung als Wildkamera ideal, da sie nicht so leicht auffällt.
- Hat eine hohe Rechenleistung, was unter anderem an dem Dual-Core-Prozessor liegt.
- Sehr günstig (11,99€ auf Amazon)

Eine Alternative ist die ESP-EYE, welche über ähnliche Fähigkeiten verfügt, jedoch ist sie mit mindestens 18,95€ eindeutig teurer als die ESP32CAM.

2.3. Arduino

Für dieses Projekt wird die IDE Arduino verwendet. Die Gründe dafür sind:

- Es ist übersichtlich und man muss nicht jedes Feature kennen, um es nutzen zu können.
- Es gibt Beispiel Codes, welche eine gute Grundstruktur bilden. Dadurch kann man sich auf die projektspezifischen Programmzeilen im Code konzentrieren.
- Ähnliches gilt für die Bibliotheken, die man auf Arduino oder im Internet für Arduino finden und einbinden kann. Diese enthalten bereits definierte Begriffe und Variablen. Vorhandenes erneut zu definieren wäre nicht hilfreich und würde dem Grundgedanken von Jugend forscht, das Erforschen und Umsetzen neuer Ideen, widersprechen.
- Arduino ist kostenlos.
- Es kann auf vielen Geräten verwendet werden, was vor allem für mich wichtig war, da ich an unterschiedlichen Geräten gearbeitet habe.
- Arduino ist simpel aufgebaut und somit leichter zu verwenden. Dadurch wird Zeit gespart, die benötigt werden würde, sich in eine komplexere IDE reinzulesen.
- Sie ist weit verbreitet und wird von vielen Benutzern verwendet. Das führt dazu, dass es mehr Wissen und Austausch im Internet gibt. Zudem kann man sich sicher sein, dass Arduino weitestgehend fehlerfrei ist.

2.4. Code für die ESP32CAM

Zuerst habe ich Nachforschungen angestellt und Informationen gesammelt. Dabei habe ich herausgefunden, dass es einen Beispiel Code zur Unterstützung von Projekten auf Arduino mit AI-Thinker Modellen, zu welchen auch die ESP32CAM gehört, gibt. Ich habe diesen als Grundstruktur für meinen Code verwendet. Von dieser Grundstruktur aus mussten noch einige projektspezifische Features und Veränderungen eingebracht werden. Diese umfassen unter anderem, aber nicht nur:

- Eine SoftAP (software enabled access point) einrichten
- Den Beispiel Code auf die ESP32CAM spezifizieren
- Kamera Einstellungen in den Code einbringen und verändern
- SD-Karte einbinden
- Einbau eines automatischen Bild-Auslösers
- Einbau eines Bild-Zählers

2.4.1. SoftAP

Durch die Inkludierung einer SoftAP hat die ESP32CAM ein eigenes WLAN und muss nicht zwingen mit einem WLAN-fähigen Gerät verbunden sein. Dies ist insbesondere für mein Projekt hilfreich, da die „Intelligent Wildlife Watch“ im Wald platziert wird, ohne die Möglichkeit auf ein externes WLAN zugreifen zu können.

Um eine SoftAP einzurichten, müssen zuerst die entsprechenden Bibliotheken eingebunden werden. Anschließend müssen eine SSID und ein Passwort vergeben werden. Es gab auf manchen Geräten Probleme sich in diesem Wifi anzumelden, daher waren mehrere Versuche notwendig. Dabei habe ich herausgefunden, dass es an manchen Geräten nicht möglich ist sich einzuloggen, wenn das Passwort zu lange ist. Aufgrund dessen sollte das Passwort acht Zeichen nicht überschreiten.

2.4.2. Beispiel Code auf die ESP32CAM spezifizieren

Da der Beispiel Code für mehrere AI-Thinker Modelle gedacht ist, musste zuerst definiert werden, welches das genutzte ist. Danach ist es hilfreich alle Befehle zu den anderen Modellen zu entfernen, denn dadurch wird der Code übersichtlicher und somit weniger anfällig für Fehler.

2.4.3. SD-Karte einbinden

Da mein Projekt Aufnahmen verwendet, braucht es viel Speicherplatz. Aufgrund dessen wurde eine SD-Karte hinzugefügt.

Im Standard ist nicht vorgesehen, dass die aufgenommenen Bilder auf einer externen Karte gespeichert werden. Dies wurde programmtechnisch implementiert.

2.4.4. Automatischer Auslöser

Im Code ist der automatische Auslöser eine bereits definierte Funktionalität unter dem Namen „Delay Time“. Diese wird in Millisekunden angegeben und entscheidet nach welcher Zeitspanne ein Bild erstellt wird.

2.4.5. Bild Zähler

Der Bild-Zähler nummeriert die Bilder, die gemacht wurden, und zeigt im seriellen Monitor die Nummerierung an. Dadurch ist es einfacher den Überblick zu behalten. Außerdem hilft die Nummerierung beim Finden von einzelnen Bildern. Sobald die ESP32CAM aus- und wieder eingesteckt wird, fängt der Bild-Zähler wieder bei null an und überschreibt die alten Fotos mit den neuen. Bei Bedarf kann jedoch auch die Anzahl an Bildern, die bereits aufgenommen wurden, als Startzahl verwendet werden. Dadurch werden die bereits aufgenommenen Bilder nicht überschrieben. Die Startzahl ist einfach die Zahl, die mit dem Bild-Zähler im Code gleichgesetzt ist.

Der Bild-Zähler ist, wie der automatische Auslöser, eine bereits definierte Funktionalität.

2.4.6. Einstellungen für die ESP32CAM

Um qualitativ gute Bilder zu erhalten, mussten einige Einstellungen, wie die Helligkeit und Qualität verändert werden. Um an die Einstellungen zu kommen und das Live-Bild der ESP32CAM sehen zu können war es nötig, nach dem Kompilieren, noch einige Schritte durchzuführen. Zuerst öffnete ich unter Werkzeuge (bei Arduino) den seriellen Monitor. Anschließend musste der Reset-Knopf an der ESP32CAM gedrückt werden. Dadurch wird beim seriellen Monitor eine IP angezeigt. Diese muss in einem Browser eingeben werden, jedoch muss man sich zuvor mit dem WLAN verbinden, welches im Code implementiert wurde. Wie bereits erwähnt ist zu beachten, dass entsprechend des Computer Modells, das Passwort nicht über acht Zeichen lang sein darf. Dies wurde mir durch mehrere Versuche an

unterschiedlichen Geräten bewusst. Daher wurden einfachheitshalber die Zahlen eins bis acht als Passwort definiert. Nach dem Öffnen der Website, können die Einstellungen, sowie die Live-Übertragung der ESP32CAM gesehen werden.

Da die Bilder trotz starker Beleuchtung sehr dunkel waren, musste die Helligkeit auf die höchste Stufe (vier) gestellt werden. Außerdem musste auch die Kameraqualität reduziert werden, da ansonsten grüne Streifen auf den Bildern waren. Zudem wurde die LED-Helligkeit auf Maximum erhöht (256).

2.5. Bilder aufnehmen

Eine KI benötigt viele Daten, damit sie eine möglichst hohe Trefferquote erreicht. Dies ist vor allem bei der Objekterkennung wichtig. Es gibt drei Wege Daten bzw. Bilder zu sammeln:

- Bilder aus dem Internet nehmen. Dabei muss darauf geachtete werden, dass diese nicht urheberrechtlich geschützt sind.
- In der Galerie bereits vorhandene Bilder nutzen.
- Sie selbst aufnehmen.

Alle drei Wege haben ihre Vor- und Nachteile. Ich habe mich für die dritte Option entschieden. Zum einem habe ich in meiner Galerie nicht die passenden Bilder. Zum anderem könnte ich es offensichtlich nicht vorführen, hätte ich mit Fotos von echten Tieren gearbeitet. Daher nutze ich für den Prototypen Bilder, die ich selbst aufnehme, und zwar von Spielzeugtieren. Bei der Methode meiner Wahl gibt es auch wieder unterschiedliche Optionen, wie es umgesetzt werden kann:

- Mit der Web-CAM eines Computers
- Mit einer Handykamera
- Mit einem Kameramodul, in meinem Fall das OV2640 Modul bzw. die ESP32CAM

Natürlich nutze ich die dritte Option, da ich extra dafür den Code geschrieben und Einstellungen verändert habe (siehe „Einstellungen für die ESP32CAM“). Durch den automatischen Auslöser wurde alle 10 Sekunden automatisch ein Bild aufgenommen.

Der Hintergrund muss neutral sein, damit die KI die Tiere nicht anhand von Objekten oder Farben im Hintergrund erkennt. Aufgrund dessen habe ich einen komplett weißen Ort verwendet. Zudem müssen möglichst alle Schatten eliminiert werde, da die KI sie ansonsten als Anhaltspunkte verwenden könnte. Deswegen habe ich mir ein professionelles Lampen Setup geliehen und aufgebaut.

Um zu schauen, wie die Bildqualität ist, habe ich den Livestream die ganze Zeit angehabt und immer wieder überprüft, ob es irgendwelche Störfaktoren gibt.

Die Bilder von den Objekten müssen aus jeder vorher definierten Entfernung, Position und Richtung gemacht werden. Um sicher zu stellen, dass ich diese Rahmenbedingungen einhalte, habe ich bei allen Tieren eine vorher definierte Reihenfolge abgearbeitet.

Zuerst habe ich die Tiere in eine ungefähr mittlere Entfernungsposition gestellt. Das Tier hatte immer zuerst den Kopf Richtung Kamera gerichtet. Damit die KI sicherer bei ihren Ergebnissen ist, habe ich mehrere Bilder von den Objekten in der gleichen Position gemacht. Um dies zu realisieren habe ich mir einen 45 Sekunden Timer gestellt, wodurch ich fünf Bilder pro Position hatte. Meine Wahl fiel auf 45 Sekunden statt 50 Sekunden um immer bereit zu sein das Tier, direkt nach dem ein Foto gemacht wurde (was ich am LED-Licht sah), zu bewegen. Ansonsten wäre meine Hand immer wieder drauf gewesen und ich hätte doppelte Arbeit gehabt die ganzen Bilder, auf denen meine Hand zu sehen ist, zu löschen. Anschließend habe ich das Tier um ungefähr 45° Grad gedreht, also 8-mal je Entfernungsposition. Das heißt, dass die Kamera bzw. die ESP32CAM an einer festen Position war und nicht bewegt wurde. Als Nächstes habe ich das Tier weit weggestellt, wieder die Bilder gemacht und dasselbe anschließend mit einer nahen Position durchgeführt. Dann habe ich Bilder von den Tieren weit rechts oder weit links und dabei auch in unterschiedlichen Entfernungen, sowie Positionen gemacht. Jedoch weniger als zuvor. Der darauffolgende Schritt war es die ESP32CAM auszustecken, die Speicherkarte herauszunehmen und in den Computer zu stecken. Anschließend habe ich mir die Bilder angeschaut und schlechte bzw. unscharfe gelöscht. Weiter habe ich den Ordner zum entsprechenden Tier umbenannt (dies ist später für Edge Impulse hilfreich) und ihn auf den PC gezogen. Der nächste Schritt bestand in dem Beschriften (siehe „Bilder beschriften“).

Dann habe ich den Vorgang für die zwei anderen Tiere wiederholt, inklusive dem Übertragen auf den PC, die Umbenennung des Ordners und das Beschriften. Sobald ich zum nächsten Tier übergegangen bin bzw. die ESP32CAM aus- und eingesteckt habe, wurden die alten Bilder absichtlich überschrieben. Das habe ich in meinem Code mit dem Bild-Zähler festgelegt. Hätte ich gewollt, dass die Bilder nicht überschrieben werden, hätte ich ihn nur auf so viele Bilder ändern müssen, wie ich bereits gemacht hatte. Jedoch war das nicht nötig, da ich die anderen Bilder schon übertragen hatte und sie auf der Speicherkarte nicht mehr benötigte. Im Prinzip habe ich mir dadurch das Löschen gespart.

Letztendlich bin ich auf durchschnittlich 160 Bilder pro Tier gekommen.

Zusätzlich habe ich zehn Bilder von dem Hintergrund ohne ein Objekt bzw. Tier gemacht. Auf dem Computer habe ich diese dann in einen eigenen Ordner mit dem Namen „Hintergrund“ geschoben.

2.6. Edge Impulse

Edge Impulse ist eine Plattform für die Entwicklung und Implementierung von KI-Modellen. Diese wurde von dem niederländischen Unternehmen Edge Impulse B.V. gegründet. Ich habe mich für diese Plattform entschieden, da sie im Vergleich zu anderen Plattformen leistungsstärkere und gleichzeitig ressourceneffizientere Modelle ermöglicht.

Edge Impulse ist nicht nur kostenlos, sondern verlangt außer einer E-Mail-Adresse keine persönlichen Daten zur Anmeldung.

Um sie nutzen zu können, muss man sich nur einen Account erstellen. Nach der Accounterstellung gelangt man auf die Startseite. Dort gibt es viele Features. Für mein Projekt sind erst einmal nur „Data Acquisition“ und „Impulse Design“ relevant. Unter „Impulse Design“ wird eine Unterkategorie angezeigt, mit dem Namen „Impuls erstellen“. Nach einigen weiteren Schritten werden zwei weitere Unterpunkte angezeigt, „Bild“ und „Objekterkennung“. Zum Schluss wird ein weiteres Feature für mein Projekt relevant, und zwar „Deployment“. Wofür die Features benötigt werden und was sie machen, wird unter anderem in den nächsten drei Unterkapiteln erklärt. Mittig auf der Startseite ist ein Feld mit dem Namen „Add existing Data“. Hier werden die zuvor aufgenommenen Bilder hochgeladen.

2.6.1. Bilder beschriften

Der erste Schritt ist die aufgenommenen Bilder hochzuladen. Dafür muss „Data Acquisition“, auf der Startseite von Edge Impulse ausgewählt werden. Auf der dann geladenen Webseite werden die Dateien bzw. Bilder hochgeladen. Ich habe diese bereits im Vorhinein in Ordner sortiert und lade zuerst nur einen hoch. Das bedeutet, dass ich erstmal nur die Bilder von einem der drei Tiere beschrifte. Da ich den Ordnern bereits zuvor Namen gegeben habe, wird dieser direkt für die Bilder übernommen. Wichtig ist dies nicht zu verwechseln mit dem Beschriften der Objekte. Der Name sagt nur aus, wie die Datei heißt, nicht das Objekt. Das Beschriften der Objekte kommt später. Weiter muss jetzt entschieden werden, ob die Bilder für die Tests der KI, das Training oder beides sind. Ich habe die Bilder für beides genutzt.

Die jetzt hochgeladenen Bilder befinden sich in der sogenannten „Labelingqueue“. Die „Labelingqueue“ ist die Warteschlange für Bilder, die noch keine Beschriftung haben. Wie funktioniert das „Beschriften“? Es gibt eine „Bounding box“ auf Deutsch einen Begrenzungsrahmen. Dieser muss manuell an die linke obere Ecke, des auf dem Bild zusehendem Objekt, gesetzt werden. Im Weiteren müssen dann die restlichen Seiten des Rahmens angepasst werden, so dass alles vom dem Objekt von dem Rahmen umschlossen wird. Als nächstes muss er beschriftet werden. Die Beschriftung sollte der Name des Objekts sein, denn dieser wird auch später bei der Objekterkennung angezeigt. Dieser manuelle Aufwand war für alle der ca. 460 verwendeten Bilder notwendig. Die Beschriftung bleibt, bis sie verändert wird, für den Rahmen auf jedem Bild identisch. Aus diesem Grund wäre es von Nachteil, wenn sich mehrere Tiere in der „Labelingqueue“ vermischen, denn dann müsste der Name immer wieder verändert werden. Deswegen habe ich immer nur die Bilder von einem Tier zur gleichen Zeit hochgeladen. Gleichzeitig löschte ich alle unscharfen oder schlecht zu erkennenden Bilder.

Alle Schritte inklusive dem Hochladen mussten für jedes der drei Tiere wiederholt werden, mit dem einzigen Unterschied, dass die Beschriftung immer anders sein musste.

Jedoch gab es noch einen Ordner, nämlich der mit den Hintergrundbildern. Der Hintergrund soll nicht als eigenes Objekt erkannt werden. Er ist nur dafür da, um sicherzustellen, dass die KI eines der Tiere nicht am Hintergrund fest macht. Diese Bilder müssen einfach in der „Labelingqueue“ übersprungen werden. Das heißt, dass sie keinen Rahmen kriegen und somit auch nicht beschriftet sind.

Zum Abschluss ist es nochmal möglich sich die Bilder anzuschauen und zu löschen. Dasselbe gilt für die Rahmen mit ihrer Beschriftung. Nachdem ich nochmal drüber geschaut hatte, folgte der nächste Schritt, das Trainieren der KI.

2.6.2. KI trainieren

Das Verhältnis zwischen Trainieren und Testen bei der KI ist standartmäßig 80/20.

Der erste Schritt auf Edge Impulse ist es einen „Trainings Impuls“ zu designen. Dies machte ich in dem ich auf „Create Impulse“ gehe. Dort befinden sich mehrere Felder. Als erstes wird eingestellt, wie breit und hoch die Bilder sein müssen. Für diese Angaben gehe ich der Empfehlung von Arduino nach und nutze den Wert 48 für die Breite und die Höhe. Als nächstes muss ein „Bilderverarbeitungsblock“ hinzugefügt werden. Dieser verarbeitet und passt die Bilder so an, dass sie zum Training der KI verwendet werden können. Dazu gehören

Veränderungen wie z.B. die Anpassung der Größe, der Kontraste oder die Skalierung der Pixelwerte. Dies führt zu einem besseren Lernerfolg der KI. Nun wird ein Lernblock eingefügt. Dieser definiert, was die KI überhaupt lernen soll und muss immer entsprechend den Daten sein, die sie zuvor erhalten hat. Für mich gibt es natürlich nur die Option „Objekterkennung“.

Ganz links werden einem jetzt die Features angezeigt, die die KI haben wird, in meinem Fall die Erkennung von Wölfen, Wildschweinen und Rehen. Nachdem ich den Impuls gespeichert habe, sieht man die zwei weitere Optionen am linken Rand, von denen ich eben geschrieben habe („Bild“ und „Objekterkennung“). Dies sind die zwei Blöcke, die ich zuvor ausgewählt habe.

Unter „Bild“ muss noch ausgewählt werden, welche Farbtiefe die Bilder haben sollen, die ich hochgeladen habe. Entweder RGB oder Grayscale. RGB heißt, dass jedes Pixel in der Regel entweder Rot, Grün oder Blau ist und Grayscale, dass alle Pixel einen Grauton haben. Aufgrund der niedrigen Qualität der Bilder, sind meine eher Grayscale.

Nach dem Speichern der ausgewählten Parameter erscheint die nächste Webseite.

Zuvor wird auf der rechten Seite angezeigt, wie lange und wie viel RAM benötigt wird, um ein Bild zu verarbeiten.

Auf der jetzigen Webseite wird einem nochmal angezeigt, wie viele Bilder und Features genutzt werden. Nachdem man „Features laden“ drückt, wird einem angezeigt, wie gut die KI die Tiere unterschieden hat. Es gibt einen Graphen, auf dem die Bilder als Punkt angezeigt werden. Im besten Fall sind alle Bilder von einem Tier zusammen in einer Gruppe, mit Abstand zu den Bildern von den anderen Tieren.

Jetzt geht es weiter zu der Kategorie „Objekterkennung“. Dort kann eingestellt werden, wie viele Trainingszyklen es geben soll und die Lernrate der KI.

Der Trainingszyklus sagt aus, wie oft die KI das Training wiederholen soll und die Lernrate wie viel die KI aus den Fehlern lernen soll.

Bei einer niedrigen Lernrate ist die Anpassung der Objekterkennung genauer, jedoch benötigt das Training auch entsprechend mehr Iterationen.

Ich nutze wieder die Zahlen die Arduino empfiehlt, und zwar 30 Trainingszyklen und eine Lernrate von 0,005.

Bei Problemen mit der „Objekterkennung“ können diese Parameter immer noch angepasst werden.

Als Letztes trainieren wir die KI und erhalten die Ergebnisse. Diese zeigen an, wie akkurat die Objekterkennung war und welche Tiere miteinander vertauscht wurden und wie oft.

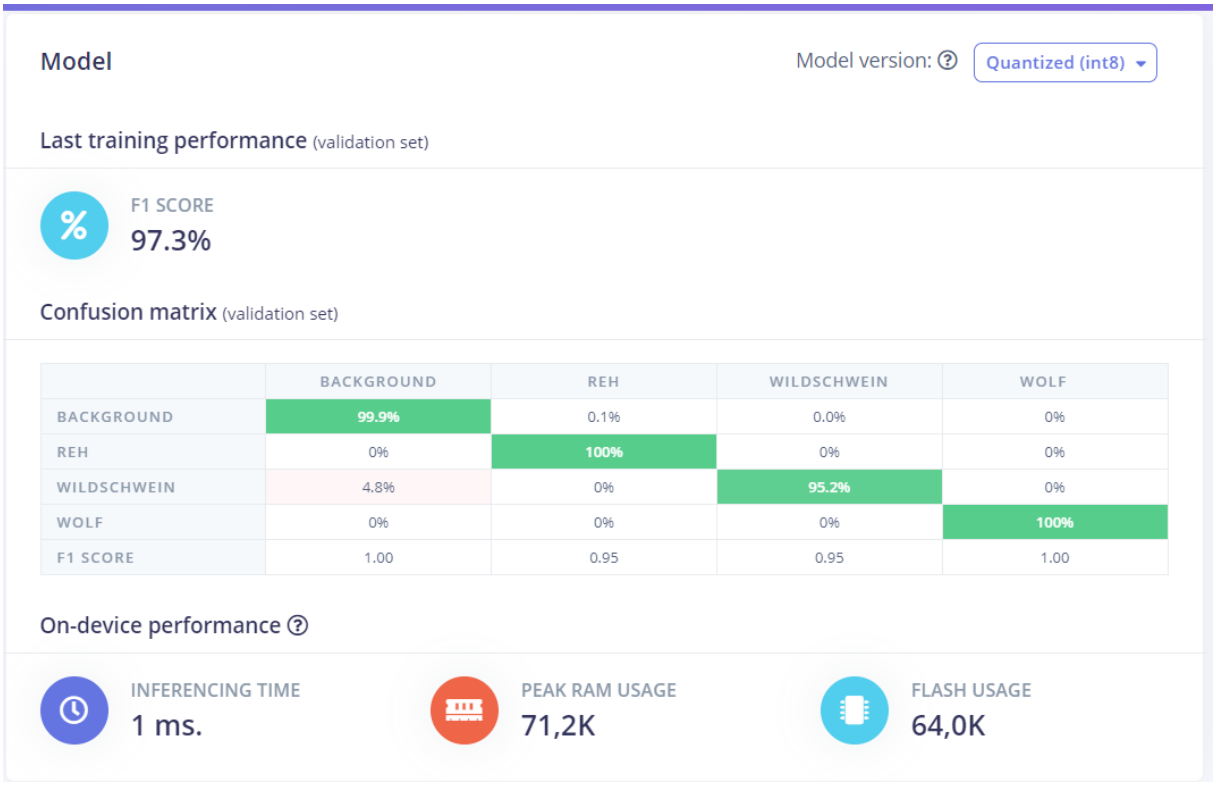

Abbildung 3:Trainingsergebnisse

2.6.3. Übertragung auf Arduino

Im letzten Schritt wird der auf Edge Impulse generierte Code auf Arduino übertragen.

Dafür muss auf Deployment gegangen werden und dort die Option „nach Arduino“ ausgewählt werden.

Edge Impulse erstellt daraufhin einen Beispielcode, den man so implementieren und nutzen kann.

2.7. Hülle

Wie ich am Anfang bereits erwähnt habe, ist eine Kamera kaputt gegangen, weil ich keine adäquate Hülle hatte. Aus diesem Grund hatte ich mir Gedanken gemacht, wie ich an eine passende Hülle für die ESP32CAM komme. Nach längerer Überlegung kam ich zu dem Schluss, dass es am sinnvollsten wäre, selbst eine zu designen. Die Hülle musste mehrere Anforderungen erfüllen:

- Die ESP32CAM darf nicht komplett umhüllt sein, weil sie ansonsten überhitzt
- Sie muss abnehmbar sein
- Es muss eine Lücke für den USB-Anschluss geben
- Der Resetknopf an der ESP32CAM muss noch erreichbar sein
- Die Hülle muss ein Loch für die Kamera haben

Ich entschied mich für das Programm FreeCAD, welches eine App zur 3D Modellierung ist.

Damit die ESP32CAM nicht überhitzt, habe ich oben und unten bis auf kleine Ränder alles komplett freigelassen. Dort wo sich der USB-Anschluss befindet, habe ich auch diesen weg gelassen, so dass der Anschluss erreichbar bleibt. Weiterhin habe ich vorne ein Loch für die Kamera frei gelassen. Die Seiten habe ich etwas länger gezogen und dann dort kleine Vertiefungen gemacht. Als Rückseite wird eine Platte verwendet, die in die Vertiefungen reingeschoben wird.

Dann habe ich die Maße der ESP32CAM genommen und meine Überlegungen in die Tat umgesetzt, indem ich diese in FreeCAD designend habe. Anschließend habe ich die Hülle mit einem 3D Drucker gedruckt.

3. Zukünftige Pläne

Das größte Vorhaben, welches ich noch machen möchte bzw. welche in Arbeit ist, wäre eine Webseite zu erstellen. Diese wird mit der IDE Visual Studios erstellt.

Dort sollen sich die Besitzer einer „Intelligent Wildlife Watch“ anmelden und auf ihre Kamera zugreifen können. Weiter sollen dort die Benachrichtigungen ankommen, dass ein Tier gesichtet wurde und welches Tier. Sollte ein Benutzer mehrere „Intelligent Wildlife Watch“ haben, wird dort angezeigt, welche der Kameras ein Tier entdeckt hat. Dadurch weiß die Person dann auch den Ort.

Zudem wird es auf ihr ein automatisches Zählsystem geben.

4. Literaturverzeichnis

Amazon. (kein Datum). Amazon. Von IR-LED-Platinen Modul: https://www.amazon.de/POFET-Infrarot-IR-LED-850-nm-Platinenmodul-f%C3%BCr-das-Kamera%C3%BCberwachungssystem/dp/B087279ZJL/ref=asc_df_B087279ZJL/?tag=googshopde-21&linkCode=df0&hvadid=447522061084&hvpos=&hvnetw=g&hvrand=8627532907961185447&hvpone=&hvptwo=& abgerufen

GEO. (2021). Von https://www.geo.de/natur/tierwelt/immer-mehr-woelfe-werden-ueberfahren-30494188.html abgerufen