Gépi tanuláson alapuló objektumdetektálási és –klasszifikálási funkció megvalósítása Raspberry Pi-vel

Szilák Máté*. Aradi Szilárd**

**Budapesti Műszaki és Gazdaságtudományi Egyetem, MSc hallgató,*
(e-mail: szilak.mate@gmail.com).
*** Budapesti Műszaki és Gazdaságtudományi Egyetem, egyetemi adjunktus,*
(e-mail: aradi.szilard@mail.bme.hu)

Absztrakt: Napjainkban a gépi tanulás egyre több műszaki területen tölt be fontos szerepet. Ez különösen igaz a képfeldolgozási technológiákra, ahol az objektumok detektálása és klasszifikálása ma már elképzelhetetlen lenne gépi tanulás és mesterséges neurális hálózatok nélkül. Azonban ezek az algoritmusok mind a tanulási, mind pedig az előhívási fázisban jelentős számítási kapacitást igényelnek. Fontos azonban megjegyezni, hogy a műveletek jelentős része mátrixos formában felírhatók, így speciális hardverekkel ezek a számítások könnyen gyorsíthatók. Kiváló példa erre a GPU-k alkalmazásának elterjedése a gépi tanulás területén. Problémát jelent azonban még a beágyazott rendszereken, vagy az azokat helyettesítő alacsony költségű és alacsony számítási kapacitással rendelkező miniszámítógépeken történő futtatás. Ez a cikk bemutatja, hogy milyen előnyökkel jár egy külső hardveres gyorsító amennyiben képfeldolgozást szeretnénk végezni valós időben egy kisteljesítményű miniszámítógépen, illetve egy példát mutat annak felhasználására.

1. BEVEZETÉS ÉS MOTIVÁCIÓ

Az elmúlt években egyre inkább előtérbe kerültek a kamerás képfelismerő rendszerek a járműiparban. Egy fontos alapfeladat a környezet objektumainak felismerése és klasszifikálása. Számtalan funkció épül erre a táblafelismeréstől az automatikus vészfékasszisztensig.

A BME Közlekedés- és Járműirányítási Tanszéken hallgatók egy csoportja egy önvezető elektromos gokart fejlesztésén dolgozik. Cikkünkben egy arra készült képfeldolgozási funkciót mutatunk be. A cél egy olyan modul megalkotása, mely képes a modul előtt látható közlekedésben résztvevő objektumok típusára (pl: kerékpár, gyalogos, személygépjármű, autóbusz), és a felismert objektumok távolságára vonatkozó információkat adni, továbbá képes a sebességvektoruk meghatározására is közel valós időben.

A tervek szerint a modul két különálló egység szenzorfúziójából áll össze, egy autóipari radarból és egy képfeldolgozást végző egységből. Ez a cikk a képfeldolgozást végző modult mutatja be, annak hardveres és szoftveres megvalósításával.

A cikk második fejezete az objektumfelismerő mesterésges neurális hálózatok működését tárgyalja, a harmadik fejezetben bemutatjuk a felhasznált hardvereket, majd a negyedik fejezeben a kifejlesztett szoftvert és a felhasznált modulokat. Végül az ötödik fejezet ismerteti a teszteredményeket hardveres gyorsítás használatával és anélkül, majd a hatodik fejezetben az eddig elvégzett munka konklúziójával zárjuk cikkünket.

2. OBJEKTUMFELISMERŐ NEURÁLIS HÁLÓZATOK

2.1 Gépi látás

A gépi látás az informatika egyik ága, mely azzal foglalkozik, hogy a számítógépek képesek legyen látni, azonosítani és feldolgozni képeket úgy, ahogyan az emberek képesek. Gépi tanuló algoritmusokkal sikerült áttörést elérniük a kutatóknak ezen a területen, hogy a számítógépek képesek legyenek értelmezni a képeket, videókat. Alapvetően az áttörést, hogy a gépi látás manapság már használva van az iparban, 2 fő kritikus dologra tudjuk visszavezetni:

Manapság lehetséges hozzáférni megfelelő teljesítményű videokártyákhoz (GPU – graphical processing unit) megfizethető áron, ugyanis az elmúlt 20 évben a videokártyákkal szemben támasztott igények fokozatosan nőttek (elsősorban a számítógépesjátékok által támasztott igények által), és tömeggyártásban értelemszerűen nagyságrendekkel olcsóbban lehet hozzájutni ilyen hardverekhez. Azért ez a hardver van kiemelve, mert ezen lehetséges párhuzamosan futtatni több szoftvert, és ezen értek el nagyon jó eredményeket.

Big Data, a cloud storage, a közösségi oldalak elterjedésével nagyságrenddel több adat áll rendelkezésre az egyes kutatóknak és cégeknek, ezáltal nagyobb adathalmazon tudnak fejleszteni és tesztelni, és mint ahogy a további pontokból kiderül, a gépi tanulásnál kardinális kérdés a rendelkezésre álló tanulóadatok mennyisége. [1]

2.2 Gépi tanulás

A gépi tanulás a mesterséges intelligencia része, mely olyan algoritmusokkal foglalkozik, mely képes egy adathalmazból tanulni, anélkül, hogy direktbe beprogramoznánk, és utána képes ismeretlen adatokra jósolni, következtetéseket levonni.

Három fajtát különböztetünk meg az algoritmus betanításától függően, melyeket az alábbiakban ismertetünk.

Felügyelt tanulás

Az algoritmus tanításához használt adathalmaz fel van címkézve. Például ha egy olyan programot szeretnénk, ami felismeri a kézzel írt számjegyeket, akkor minden egyes számjegyhez tartozik több ezer kép, mely azt a számot ábrázolja.

Nem felügyelt tanulás

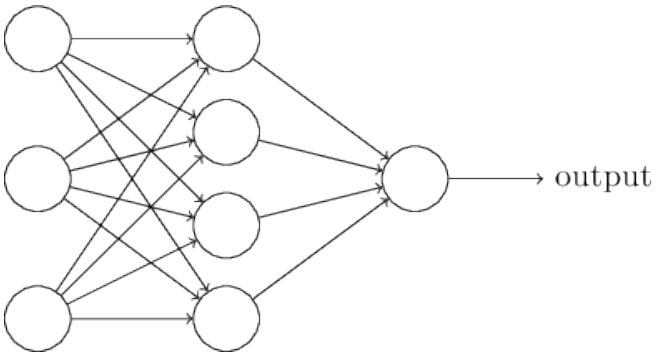
A nem felügyelt tanulásnál nem áll rendelkezésre címke az adatokhoz. Itt a „clusterezés" a feladat, tehát csoportokra osztani az adatot hasonlóság alapján. Ez például akkor hasznos, hogyha egy adatbázist akarunk megérteni.

Megerősítéses tanulás

Ez egy olyan tanulási módszer, amikor a környezettel kapcsolatba lépve „trial and error" módon tanul. Ez magyarul azt jelenti, hogy van egy cél, amit megkap a hálózat, és a környezettel kapcsolatba lépve kudarc / siker alapján tanul be. Ez lehetővé teszi az algoritmusnak, hogy beállítsa az ideális viselkedést.[2]

2.3 Neurális hálózatok

A képfelismerő algoritmusok között az ún. mélytanuló (deep learning) algoritmusok terjedtek el. Ezek több szintű ún neuronok hálózata. Egy hálózatnak van n számú bemenete, és k számú kimenete, és közöttük vannak a neuronok.



1. ábra: A mesterséges neurális hálózat felépítése

Az 1. ábrán egy olyan neurális hálózat látható, melynek 3 bemenete és 1 kimenete van. 1 layer (szint) található benne, és 4 neuron. Úgy működik, hogy a bemenetet megkapják egy aktivációs függvénnyel a neuronok. Az aktivációs függvény értéke [0;1] intervallumba esik, és sigmoid tipusú, és ez dönti el, hogy mekkora erősítése lesz a bemenő információnak. A neuronok összeadják a kapott értékeket, és végül így jutunk el a kimenetig.

A konvolúciós neurális hálózatok olyan hálózatok (CNN – convolutional neural network) melyeknél legalább egy layer konvolúciós, a korábban bemutatott neurális hálózatokba illeszthető. Működését tekintve abban különböznek, hogy az egyes aktivációs függvények mellett filterek, szűrők vannak. Ezeket mátrixként tudjuk elképzelni, és tartalmazza egy mintázatra jellemző objektumot, amire szűr, pl lehetnek ezek élek, körök, négyzetek. Így egy képet sok kicsi apró jellemzőre bontunk le, és minden egyes kimenetnek megvannak a saját jellemző kimenetei, és minél több apró jellemző stimmel, a kimeneten annál nagyobb értéket kap az adott kimenet fajta. [3]

2.4 Objektum felismerő hálózatok

Ezek a konvolúciós neurális hálózatok részei, abban különböznek a hagyományos képfelismerő hálózatoktól, hogy képesek több objektumot is felismerni egy képen. Számunkra azért csak és kizárólag az ilyen tipusú hálózatok alkalmazhatóak, mert az applikációban követelményeinek csak ez felel meg.

Több fajtája terjedt el, melyekből a legfontosabbakat az alábbiakban röviden ismertetjük.

RCNN

2000 apró szegmense bontja a képet, és azokra külön megcsinálja a klasszifikációt, majd összeveti a kimeneteket. A hátránya, hogy valós időben nem lehet használni, mert ahhoz lassú.

YOLO – you look only once

Felbontja a képet egy SxS-es hálóra, és abba rak határoló téglalapot. A téglalap méretétiterálja, hogy mikor tér vissza a legnagyobb valószínűséggel. Előnye hogy gyors, hátránya, hogy az apróbb objektumokat nem veszi észre.

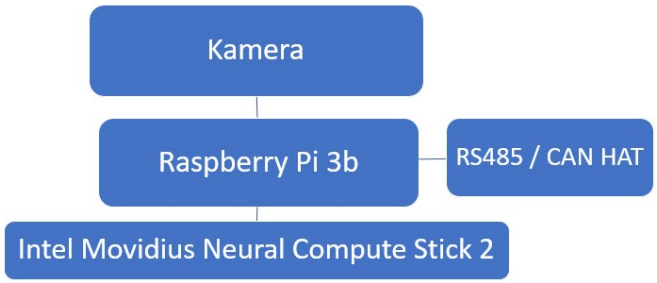
SSD - Single Shot Detector

Neurális hálózatokkal segítségével végzi a becslést, nem végez iterálást. [4]

Az [5] cikkben összehasonlították a beágyazott rendszerekre, tehát csökkentett számítási igénnyel rendelkező algoritmusokat, és konklúzióként levonhatjuk, hogy egy objektum felismerő algoritmus vagy gyors, vagy pontos, úgyhogy ezen megfontolásból a Mobilenet-SSD nevű hálózattal megyünk tovább, ami egy középút a gyorsaság és a pontosság között.

3. HARDVERARCHITEKTÚRA

Az 2. ábra mutatja be a hardveres architektúrát a képfeldolgozó egységhez.



2. ábra: Hardverarchitektúra

A központi egység egy Raspberry Pi 3 B, amelyhez a Camera Serial Interface-en (CSI) csatlakozik a Raspberry Pi kiegészítő kamerája. A CAN hálózathoz való csatlakozást egy ún. „CAN HAT" kiegészítő biztosítja, amely a Raspberry Pi 40 pólusú GPIO csatlakozójához kapcsolódik. Az Intel Neural Compute Stick hardveres gyorsító az egyik USB porton keresztül kommunikál a processzorral.

3.1 Raspberry Pi 3B

A Raspberry Pi egy bankkártya méretű miniszámítógép, melyre lehetséges monitort csatlakoztatni, billentyűzetet, egeret, és egyéb perifériákat. Nagy előnye, hogy komplex, önálló működésre képes, illetve a kis teljesítmény igény, ami lehetővé teszi a sokrétű felhasználását. Egy 5V-os, minimum 1 A-es tápról lehet üzemeltetni. A hardverre fejlesztett Linux disztribúciók futtatására képes. A Neural Compute Stick 2 (továbbiakban NCS2) USB 3.0-n keresztül fog hozzá csatlakozni. Rendelkezik GPIO lábakkal, így azokat felhasználva lehet majd illeszteni egy CAN-es hálózatra. [2]

Főbb tulajdonságai:

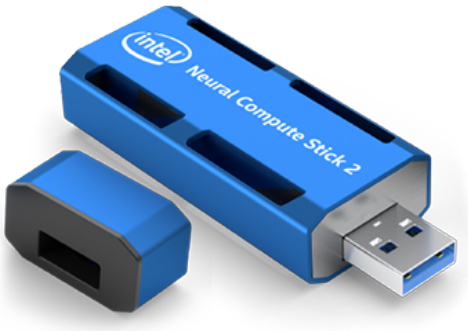
- 4 magos 1.2 GHz órajelű 64 bites processzor, Cortex-A53 (ARMv8) magokkal
- 1 GB LPDDR2 RAM
- HDMI kimenet
- Beépített WiFi és Bluetoot Low Energy
- 100 MBit Ethernet
- 40 GPIO láb
- Kamera csatlakozásra dedikált hardveres bemenet (CSI port)
- Érintőkijelző bővítés lehetőség (DSI port)
- Micro SD port

3.2 Kamera

Speciális a Raspberry Pi számára kifejlesztett kamera. A modul egy 8 MP-es felbontású Sony IMX219 szenzorra épül.

3.3 Intel Neural Compute Stick 2

Az Intel eszköze (3. ábra) egy olyan külső hardveres gyorsító, amely a neurális hálózatok számításait gyorsítja, elsősorban képfeldolgozási feladatokra használható. Az eszköz megkönnyíti beágyazott rendszerekben a neurális hálózatok felhasználását, felhasználhatjuk valós idejű applikációkban mivel nem szükséges internet kapcsolat a használathoz.



3. ábra: Intel Neural Compute Stick 2

Az eszköz az Intel Movidius Myriad X VPU-ját (Vision Processing Unit) tartalmazza és minden népszerű gépi tanulási keretrendszert (Tensorflow, Caffe stb.) támogat.

Egy USB 3-as csatlakozóval csatlakoztathatjuk a számítógépünkhöz. Továbbiakban NCS2 rövidítéssel lesz hivatkozva. [3]

3.4 RS485 / CAN HAT

Ez egy olyan kiegészítő (4. ábra), melyet a Raspberry Pi GPIO portjaira csatlakoztatva RS485 / CAN buszon tudunk kiküldeni és fogadni adatokat.

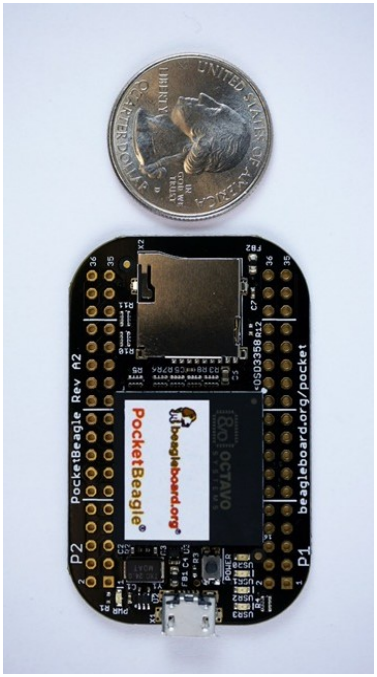
Az eszköz központi eleme egy Microchip MCP2515 typusú IC, amely SPI interfészen keresztül képes adatokat fogadni, és az abban kapott parancsoknak megfelelően képes a CAN kommunikációra.

Az eszközhöz alacsony és magas szintű szoftver csomagok elérhetőek, így implementálásuk nagyon egyszerű, ugyanis a megfelelő driverek telepítése után egy Python könyvtárral, magas szintű függvények segestégével tudunk kommunikálni a központi IC-vel.

4. ábra: RS485 CAN HAT és Raspberry Pi

3.5 PocketBeagle

A PocketBeagle (5. ábra) egy nagyon kicsi számítógép, melyet gyors prototípus készítésre tudunk használni. A rendszer központi eleme egy Octavo Systems OSD335x-SM System-in-Package (SiP). Ez egy kisméretű rendszercsip, amely az Arm Cortex-A8 architektúrájú processzor mellett tartalmaz integrált tápegységet, a legkülönbözőbb általános és ipari kommunikációs technológiákat, valamint a beágyazott rendszerekre jellemző perifériákat.



5. ábra: PocketBeagle

Itt csak néhány fontosabbat emelünk ki:

- CAN, SPI, UART, I2C, GPIO
- ADC
- USB, Ethernet, Profibus
- MMC, SD, SDIO x3

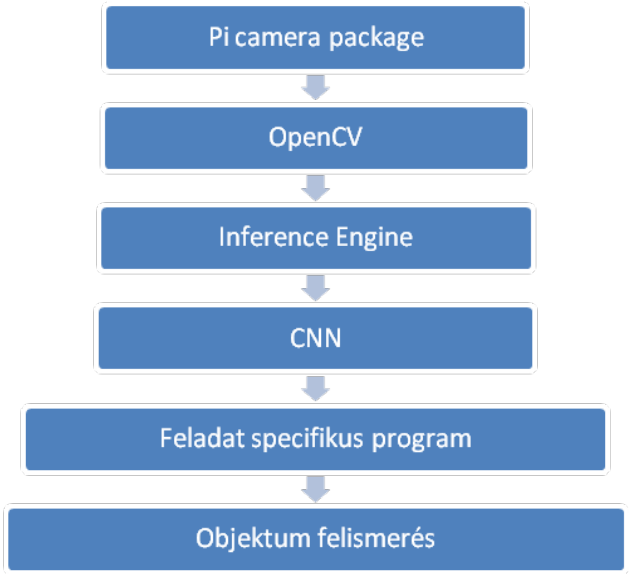
Az eszköz Debian Linux alap operációs rendszert futtat. Alapvetően USB-vel tudjuk csatlakoztatni a számítógéphez, ahol UART/SSH kapcsolatot tudunk létesíteni vele. Ahogy afenti felsorolásból is kiderült, van benne beépített CAN interface, amihez még egy szintillesztő szükséges a buszra történő csatlakoztatáshoz.

4. SZOFTVERMODULOK

A Raspberry Pi-n használt operációs rendszer egy Linux disztribúció, neve Raspbian.

A PocketBeagle-el ellentétben rendelkezik grafikus kezelőfelülettel. A Raspberry Pi gyorsasága lehetővé tette, hogy az eszközhöz monitort és billentyűzetet csatlakoztatva magán az eszközön írjam meg a programot.

A fejlesztett programokat a git verziókövető, és a GitHub szerver segítségével mentettük el és követtük.



6. ábra: Szoftverarchitektúra

A 6. ábra mutatja be az egyes program modulok függőségeit. A teljes program Pythonban lett megírva. A Pi camera package egy Raspberry Pi specifikus könyvtár, tartalmazza a kamera meghívásához szükséges alacsony szintű drivereket.

4.2 OpenCV

Ez egy gépi látással foglalkozó szoftvermodul, elérhető a számunkra releváns C++/Python nyelveken is. Rengeteg funkcionalitása van, az újabb verziókba már integrálva van az a számunkra fontos Inference Engine, tehát gyakorlatilag az OpenCV kezeli az Inference Engine API-t. A Raspberry Pi számára kiadott OpenVINO toolkit része pedig az OpenCV is.

4.3 Inference Engine

Az Intel által kiadott könyvtár. Ez tartalmazza az NCS2 gyorsító csiphez szükséges alacsony szintű drivereket. A programban ezen keresztül lehet beállítani, hogy mi legyen a „computational backend”, tehát, hogy min végezzük a számításokat. Itt kiválaszthatunk FPGA-t, videokártyát, processzort, és a külön gyorsítóchipet, jelen esetben ezzel végeztük el a kísérletet. Programozói szemszögből szemszögből ez is tulajdonképpen egy API (Application Programming Interface).

4.4 CNN

A program az Inference Engine által kiadott függvények segítségével beolvassa a képet, a megfelelő OpenCV

metódusokat felhasználva. A programban több beállítását is testre szabhatunk a felhasznált CNN-től függően. (pl: a színkódolás RGB, vagy greyscale, vagy a bemenő kép felbontását).

4.5 Feladatspecifikus program

Jelenleg itt a megvalósult feladat az, hogy mentse le a beérkező képkockákat, és mellé készítsen egy log fájt, melyben lementésre kerül a felismert objektumok típusa, pixel pozíciója, majd a valószínűsége, illetve küldje ki az utóbbi információkat CAN buszra is.

5. TESZTEREDMÉNYEK

A teszt úgy zajlott, hogy a laptop kijelzőjén ment egy forgalomban felvett videó, és egy külön monitorra volt kötve a Raspberry Pi, és ott volt megjelenítve a végeredmény, melynek eredménye a 7. ábrán látható.

7. ábra: Képernyőkép a tesztről

A teszt során a MobileNet-SSD (Single Shot MultiBox Detector) nevű neurális hálózat volt felhasználva. A MobileNet azt jelenti, hogy kisebb teljesítményű verzió, azaz beágyazott eszközökre van optimalizálva.

A Raspberry Pi processzorán futtatva **0.8 FPS**-t sikerült elérni, míg az Intel NCS2 gyorsítóchip segítségével **10.8 FPS**-re gyorsult a feldolgozás.

6. KONKLÚZIÓ

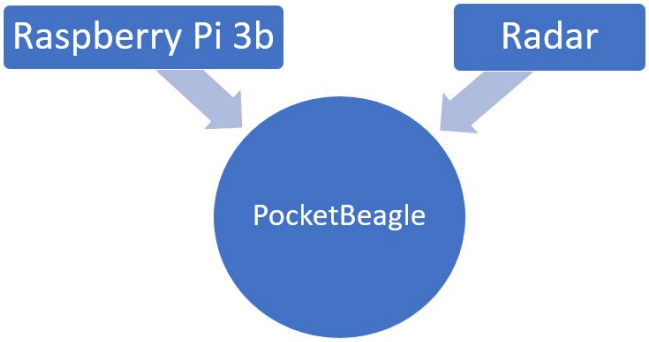
6.1 Eredmények kiértékelése

Egyértelműen kiderült, hogy szükséges egy külső dedikált hardveres gyorsítót használni a képfeldolgozáshoz, különben nem lenne alkalmas a kitűzött cél megvalósításához. A rendelkezésre álló 10.8 FPS-t még más neurális hálók kipróbálásával lehetséges, hogy feljebb lehet tornászni. A teljes szenzorfúzió kimenetének a sebessége még függ a következő fejezetben kifejtett algoritmus számítás igényétől, de alapvetően ennek a felhasználhatósága, hogy mekkora sebességig tesz lehetővé biztonságos közlekedést, a biztonsági megfontolásoktól, és a többi szenzortól függ.

Összeségében kijelenthető, hogy a kb. 10 Hz-es működése alkalmasság teszi modult az alacsony sebességű járműves kísérletek elvégzésére.

6.2 Jövőbeni fejlesztések

Amint a bevezetésben be lett mutatva, a teljes cél egy komplex szenzorfúziós modul, amely valós időben küldi ki az észrevett objektumokat és azoknak pozícióját, sebességét, irányát. A teljes konfigurációja az önvezető gokartnak még fejlesztés alatt áll. A következő lépés a gokartot szimulálva egy állványon tesztfelvételt készíteni, ahol a képfelismerő algoritmus és a radar kimenete CAN-en lementésre kerül a PocketBeagle felhasználásával. Erre azért van szükség, mert a szenzorfúziós algoritmus validálásához szükség van külső, mért adatokra, így a valós hibákkal lehet majd kitesztelni, és megalkotni.



8. ábra: Szenzorfúzió architektúra

Ezen az ábrán egy olyan teszteset látható, amivel élesben lehet mérést végezni. A teszt úgy fog megvalósulni, hogy a képfeldolgozó egység kiküldi CAN-en a felismert objektumot, annak relatív pozícióját, azaz a pixel koordinátákat, és a valószínűséget a talált objektumra. A radar szintén detektálja a környező objektumokat, és szintén CAN hálózaton kiküldi azok relatív pozícióját és sebességét. A PocketBeagle végzi a loggolást, azaz a CAN-en beérkező adatokat lementi.

A radar szenzor előnye, hogy megbízhatóbban és pontosabban képes távolságot és sebességet mérni, míg a kamerás rendszer a klasszifikálásban jeleskedik. A cél, hogy két szenzor adatait fuzionálva megbízható és pontos információink legyenek az jármű előtti térben lévő dinamikus objektumokról.

7. KÖSZÖNETNYILVÁNÍTÁS

EFOP-3.6.3-VEKOP-16-2017-00001: Tehetséggondozás és kutatói utánpótlás fejlesztése autonóm járműirányítási technológiák területén - A projekt a Magyar Állam és az

Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósul meg.

HIVATKOZÁSOK

[1] https://www.expertsystem.com/machine-learning-definition/

[2] https://www.techopedia.com/definition/32309/computer-vision

[3] https://medium.com/machine-learning-for-humans/why-machine-learning-matters-6164faf1df12

[4] https://towardsdatascience.com/beginners-guide-to-object-detection-algorithms-6620fb31c375

[5] https://www.groundai.com/project/a-comparison-of-embedded-deep-learning-methods-for-person-detection/?fbclid=IwAR0f7P0mnM4aZrDPlbFIFWSSGg5eRsLrYGnk8QTxIDeQnwXYzVOmuPZITnA