MINISTERUL EDUCAȚIEI ȘI CERCETĂRII

UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE
DEPARTAMENTUL CALCULATOARE

SISTEM DE VERIFICARE FACIALĂ PENTRU PROTECȚIA AUTOVEHICULELOR

LUCRARE DE LICENȚĂ

Absolvent: **Luciana MOȘILĂ**

Coordonator științific: **Sl. dr. ing. Vlad Cristian MICLEA**

2025

Cuprins

Capitolul 1. Introducere

În condițiile actuale ale erei digitalizării și ale dezvoltării sistemelor inteligente, tehnologiile de verificare facială devin din ce în ce mai importante în domenii vaste, inclusiv în sectorul intervențiilor de urgență. În această lucrare abordez un plan de elaborare și dezvoltare unui sistem de verificare facială pentru autospecialele de intervenții, având la bază o motivație personală și practică legată de activitatea structurilor de intervenție.

1.1. Motivația alegerii temei

Motivația alegerii acestei teme pornește de la o problemă reală identificată în activitatea echipajelor de intervenție. În cercul apropiat al meu există persoane care lucrează în domeniul gestionării situațiilor de urgență, iar în urma discuțiilor am constatat faptul că în această profesie există anumite situații care sunt dificil de gestionat. Am aflat că autospecialele de intervenție dețin aparatură electronică care necesită încărcare permanentă (defibrilatoare semi-automate, injectomate, aspiratoare electrice etc.) iar în anumite situații acest lucru vine în contradicție cu siguranța și securitatea autospecialei.

Când autospeciala este garată în unitate, alimentarea electrică se realizează printr-o sursă externă de curent, dar în timpul intervențiilor, alimentarea se face cu ajutorul motorului termic al autospecialei. Motorul generează o sursă de tensiune care menține încărcarea posibilă, dar trebuie să rămână în permanență pornit. De exemplu, pompierii sunt obligați să lase autospeciala în funcțiune, chiar și atunci când se depărtează de aceasta, pentru a evita oprirea/descărcarea aparaturii sau imposibilitatea repornirii vehiculului (descărcarea bateriei autospecialei).

Multe instituții, printre care și poliția încă are probleme ce țin de protejarea sistemelor proprii. În momentul de față, încă se utilizează sisteme învechite ce țin de protecția echipamentelor proprii. Cel mai des întâlnită este supravegherea de către personalul responsabil de automobilul sau autospeciala respectivă. Cu toate că mașinile de poliție beneficiază de echipamente foarte scumpe, acestea nu beneficiază de un sistem de protecție împotriva furtului, cum ar fi, un sistem de verificare facială ori autentificare biometrică, care să fie implementată în rândul polițiștilor care folosesc autospeciala.

În ultimii ani, s-au întâmplat suficiente cazuri în care autospecialele de poliție au fost folosite de persoane neautorizate. Aceste incidente au apărut în special în timpul misiunilor de ordine publică, când polițiștii erau angajați în gestionarea situațiilor și nu supravegheau autovehiculele.

Cauzele enumerate implică riscuri de securitate și acces neautorizat care pot duce la lucruri foarte periculoase pentru personalul propriu al autovehiculului, dar și pentru gestionarea situației în sine.

1.2. Exemple de furturi ale vehiculelor de intervenție: neglijență și lipsa tehnologiei

În presă au fost expuse cazuri în care persoane neautorizate au reușit să conducă fără drept autospecialele de poliție, unele putând fiind implicate ulterior în accidente grave.

Aceste furturi pot produce daune mult mai mari, ținând cont de faptul că în autospecialele de poliție se pot afla armament, diverse echipamente sau sisteme de comunicații interne.

Dat fiind aceste situații, implementarea unor sisteme de verificare facială pentru a debloca și conduce autospecialele de poliție este necesară. În contextul actual, în care infractorii devin din ce în ce mai ingenioși, aceste sisteme de protecție pot să ofere securitate atât pentru instituții, cât și pentru siguranța populației.

Un caz notabil din care reiese această necesitate de introducere a sistemelor de verificare facială pentru pornirea unei autospeciale de poliție a avut loc chiar în România, mai exact în Arad, în anul 2024. În timp ce polițiștii locali se aflau în patrulare în centrul orașului Arad, într-un moment de neatenție, un cetățean s-a urcat în mașina de poliție locală și a condus-o fără a avea acest drept, lovind patru mașini civile. Acest incident a fost mediatizat de către ProTv, acesția relatând faptul că mașina de poliție a fost furată chiar în fața polițiștilor. [1]

Un alt caz care relevă această necesitate a fost semnalat la Sibiu în anul 2023, caz în care un tânăr a furat o mașină de poliție, și a aplicat amenzi de circulație altor participanți la trafic. În acest caz, Ministrul de Interne de la acea vreme, a menționat că se vor efectua anchete pentru stabilirea vinovăției, cât și pentru luarea unor măsuri care să reducă sau chiar să diminueze total aceste situații. [2].

1.3. De ce metodele tradiționale de siguranță nu sunt eficiente?

Clasicele metode precum cheia fizică, cardurile de acces sau codurile PIN, nu sunt potrivite într-un mediu operativ din cauza posibilității de pierdere a acestora, degradării sau chiar uitarea codurilor de acces. Totodată folosirea unui cod pe baza unei amprente nu este sigur deoarece personalul folosește mânuși de intervenție sau protecție. Totodată, în situații de urgență, timpul este foarte important, iar folosirea unui dispozitiv suplimentar sau introducerea unui cod poate întârzia intervenția. În plus, lăsarea vehiculului în funcțiune, fără un sistem de control al accesului, presupune vulnerabilități mari.

1.4. Domeniu de aplicare și propunerea sistemului

La nivel general, proiectul meu se încadrează în zona tehnologiilor pentru vehiculelor inteligente. Din domeniul larg mă încadrez la sub-domeniul gestionării accesului automotive, ce ţine de controlul pornirii unui autovehicul specific instituțiilor speciale, maşini de pompieri sau autospeciale de poliţie. Mai jos apare biometria la bord, unde faţa şoferului devine legitimaţie în loc de chei ori coduri. La cel mai specific nivel vorbesc despre verificare facială executată de un sistem software ce controlează un microcontroler montat în maşină. Decizia se face pe loc dacă persoana poate sau nu să plece cu vehiculul.

Propunerea mea vizează un sistem de verificare facială montat direct pe vehicul, care se bazează pe un microcontroler conectat la o cameră și un software dedicat. În momentul în care cineva intenționează să preia controlul, camera îi surprinde automat chipul și detectează prezența unei fețe. Ulterior, imaginea este comparată cu baza de date a utilizatorilor autorizați, în care administratorul poate înregistra sau elimina profiluri noi. Dacă recunoașterea e validă, se activează controlul motorului și al celorlalte echipamente,

[1] Știrile ProTV, „A furat o mașină a Poliției Locale Arad...", 27 noiembrie 2024, stirileprotv.ro.

[2] Ştirile ProTV, „Un tânăr a furat o maşină de poliţie şi a oprit şoferi în trafic.", 21 octombrie 2024, stirileprotv.ro.

permițând robotului (care simulează funcționalitatea mașinii) să execute manevrele. În caz contrar, accesul rămâne blocat.

Capitolul 2 prezintă funcționalitățile generale ale sistemului. În capitolele următoare o să analizez domeniul, o să descriu arhitectura software și mecanismele de funcționare ale aplicației. Voi spune cum am implementat fiecare componentă și modul în care acestea interacționează în sistemul general. Capitolul 6 va fi dedicat strategiei de testare și procedurilor de validare a sistemului, pentru a demonstra utilitatea proiectului prezentat.

Capitolul 2. Obiectivele proiectului

2.1. Scopul proiectului

În contextul intervențiilor de urgență, fiecare secundă este esențială. În această lucrare se propune dezvoltarea unui sistem care permite identificarea automată a personalului autorizat al unei autospeciale, fără utilizarea cheilor fizice, a cardurilor sau a unei conexiuni la internet. Prin această abordare se urmărește eficientizarea accesului la vehicul și prevenirea utilizării neautorizate. Reprezentarea de ansamblu a sistemului este ilustrată în Figura 2.1.

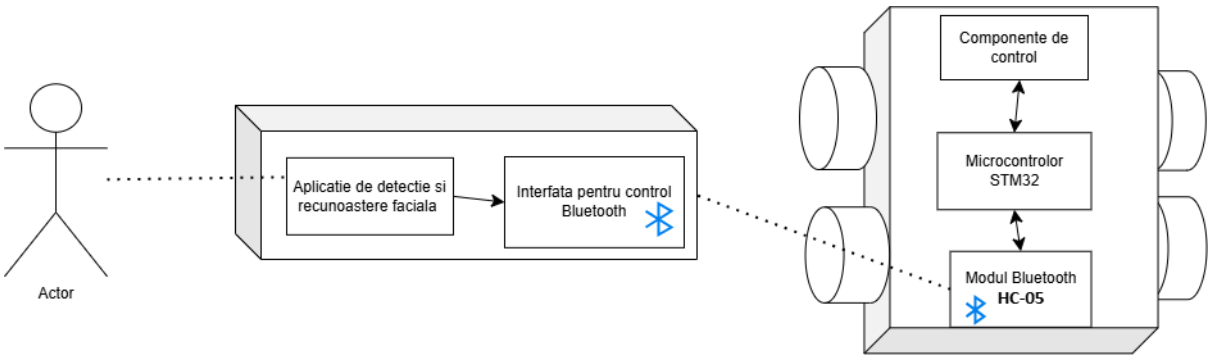


Figura 2.1: Vedere de ansamblu a arhitecturii funcționale a sistemului [1]

2.2. Obiectiv principal

Obiectivul principal al proiectului constă în realizarea unui mecanism de control al pornirii vehiculului, care să permită accesul exclusiv personalului autorizat. Se propune o soluție cu decizie locală, fără dependență de internet, cu un timp redus de răspuns pentru verificarea identității.

Obiective secundare

1. Asigurarea accesului
   Realizarea unui mecanism intuitiv prin care doar membrii echipajului au posibilitatea de a utiliza autospeciala.

2. Protejarea dotărilor
   Implementarea unor măsuri de protecție pentru componentele controlate de microcontroler, împiedicând utilizarea acestora de către persoane neautorizate.

3. Independenţă faţă de reţea
   Dezvoltarea unui flux complet: capturare, analiză, decizie; care să funcționeze fără conexiune la internet.

[1] Imagine realizată cu ajutorul platformei online `https://app.diagrams.net/`.

4. Administrare disponibilă permanent
   Introducerea posibilității de a adăuga sau elimina permisiuni în timp real, fără a opri funcționarea sistemului.

2.3. Cerinţe funcţionale

- Înscriere rapidă a utilizatorilor
  Interfața, accesibilă doar administratorului, permite încărcarea de imagini pentru fiecare profil. Salvarea acestora se realizează local, în mod automat, fără a întrerupe funcționarea sistemului. După completarea profilului, sistemul declanșează un proces de reantrenare ce durează câteva minute, pentru a încorpora noile date.

- Colectarea de date pentru antrenare
  Pentru fiecare utilizator se stochează imagini în unghiuri și condiții de iluminare variate, astfel încât modelul de recunoaștere să învețe cât mai eficient din situații reale.

- Recunoaştere în timp real
  Sistemul procesează cadrul video preluat de la cameră și oferă instant decizia privind acceptarea sau respingerea accesului.

- Detectarea chipului în cadru
  Analiza fiecărui cadru video începe doar dacă este detectată o față validă. În absența acesteia, procesul de identificare nu este declanșat.

- Decuparea automată a feţei
  După identificare, regiunea facială este decupată și redimensionată, fiind transmisă mai departe către modulul de verificare a identității.

- Prototip robot fizic
  Proiectul este validat pe un robot mobil construit fizic, care integrează un microcontroler, motoare, senzori și un șasiu cu roți. Acesta răspunde comenzilor de pornire/oprire primite prin Bluetooth, în funcție de rezultatul recunoașterii faciale.

- Pornire automată a vehiculului
  În cazul unei validări pozitive, sistemul transmite prin Bluetooth semnalul de activare a motoarelor. Dacă persoana nu este autorizată, controlul rămâne dezactivat.

- Detectarea şi evitarea obstacolelor
  Robotul este echipat cu senzori ultrasonici în față și în spate, care citesc distanța. La detectarea unui obstacol sub o anumită limită, microcontrolerul oprește alimentarea motoarelor până la eliberarea traseului.

2.4. Cerinţe nefuncţionale

- Scalabilitate
  Sistemul trebuie să rămână funcțional și stabil în cazul adăugării unui număr mare de utilizatori (de la câteva zeci la câteva sute), fără afectarea semnificativă a performanței.

- Viteză de răspuns
  Întregul proces de identificare și acționare (capturare imagine, detecție, decizie) trebuie să se realizeze în câteva secunde pentru a fi compatibil cu situațiile operative.

- Precizie şi fiabilitate
  Sistemul trebuie să distingă corect între membrii autorizați și cei neautorizați, menținând o rată scăzută de eroare inclusiv în condiții de iluminare slabă sau în medii dificile.

- Întreţinere simplă
  Gestionarea utilizatorilor (adăugare/ștergere) trebuie să se poată face ușor, direct din aplicație, fără întreruperea funcționării sistemului sau intervenții complexe asupra codului.

- Operare offline
  Aplicația de recunoaștere și control funcționează complet offline, fără dependență de servere externe sau conexiune la internet.

- Portabilitate şi extindere pe alte platforme
  Arhitectura modulară permite adaptarea sistemului și pentru alte plăci de dezvoltare (de exemplu Raspberry Pi, PC industrial), cu ajustări minime. Comunicarea dintre aplicație și dispozitivul fizic se face printr-un protocol serial standard.

În acest capitol au fost prezentate obiectivele principale și secundare ale proiectului, precum și cerințele funcționale și nefuncționale. Proiectul vizează recunoașterea automată a personalului autorizat și controlul accesului la un vehicul în mod autonom. Structura detaliată a cerințelor va ghida dezvoltarea și validarea soluției în capitolele următoare, unde vor fi analizate etapele de implementare și rezultatele obținute în scenarii reale.

Capitolul 3. Studiu bibliografic

3.1. Introducere

Recunoașterea facială a cunoscut o dezvoltare accelerată în ultimii ani, devenind una dintre cele mai răspândite metode biometrice de identificare. Este utilizată tot mai frecvent în diverse domenii, de la securitate și autentificare, până la controlul accesului, transporturi, aplicații bancare și platforme guvernamentale. În acest capitol am prezentat câteva soluții comerciale importante, dezvoltate de companii cum ar fi Neurotechnology, Regula și Kairos, am pus accentul pe aspecte tehnice și funcționale. Se analizează modul de funcționare al fiecărui sistem, nivelul de securitate pe care îl oferă, cerințele hardware necesare și flexibilitatea în scenarii concrete de utilizare.

În continuare, este descris sistemul realizat în cadrul acestei lucrări, care are drept scop recunoașterea facială pentru accesul controlat la un vehicul inteligent. Sistemul propus este analizat în paralel cu SDK-urile comerciale menționate anterior, fiind evidențiate avantajele unei arhitecturi locale, ce funcționează fără conexiune la internet și utilizează rețele siameze și componente hardware dedicate.

Următoarea parte a capitolului este dedicată unui scurt studiu bibliografic privind cele mai relevante metode moderne de recunoaștere facială. Sunt abordate în special tehnicile bazate pe învățare de tip one-shot (învățarea cu un singur exemplu), utilizarea de embedding-uri spațiale și funcții de pierdere de tip margin-based. Se discută câteva contribuții importante din literatura de specialitate, cum ar fi modelul siamez propus de Koch și colaboratorii săi, sistemul FaceNet, dar și rețele avansate precum ArcFace, care oferă o mai bună separare a fețelor în spațiul de reprezentare.

3.2. Aplicații similare

În această secțiune sunt analizate câteva soluții comerciale de recunoaștere facială utilizate pentru autentificare și control de acces. Ele sunt comparate cu sistemul realizat în cadrul acestui proiect, accentul fiind pus pe funcționalități și diferențele practice dintre ele.

3.2.1. VeriLook Face SDK – Neurotechnology

VeriLook este un SDK dezvoltat de compania lituaniană Neurotechnology. Este folosit în aplicații biometrice și sisteme de supraveghere sau control de acces. [1, 2]

Printre caracteristicile sale se numără:

- Recunoaștere facială în timp real, tolerantă la rotații ale capului până la $\pm 15°$;
- Detectare a vitalității (activă și pasivă), pentru a preveni tentativele de fraudă biometrică;
- Funcționează și în cazul purtării unei măști de protecție;
- Suportă diverse sisteme de operare – Windows, Linux, Android, iOS;
- Nu depinde de cloud – procesarea are loc local;
- Este performant cu baze mari de date.

În aplicațiile în care se cer securitate și răspuns rapid, este utilă această tehnică. Și

în aplicații proprii este ușor de implementat datorită suportului pentru mai multe limbaje de programare.

3.2.2. Regula Face SDK

Regula oferă o soluție profesională, întâlnită în domenii cum ar fi sectorul bancar sau granițele. [3, 4]

Câteva aspecte tehnice importante:

- Are detectare a vitalității certificată ISO (activă și pasivă);
- Identifică atacuri de tip deepfake, replay, măști etc.;
- Analizează calitatea imaginilor în detaliu cu peste 45 de parametri ICAO;
- Permite atât verificarea 1:1, cât și căutarea într-o bază de date (1:N).

3.2.3. Kairos Face Recognition API

Kairos este o platformă ce oferă recunoaștere facială printr-un API REST. Este potrivită pentru aplicații comerciale sau prototipuri rapide. [5, 6]

Caracteristici:

- Suportă verificare 1:1 și identificare 1:N;
- Poate estima trăsături demografice (vârstă, sex, etnie);
- Are un sistem simplu de detectare a vitalității (pasivă, din selfie);
- Funcționează în cloud sau local (on-premise), după nevoi.

Este ușor de integrat, iar modelul API este flexibil. Totuși, pentru varianta cloud e necesară o conexiune la internet, iar acuratețea depinde de calitatea camerei și de lumină.

3.2.4. Sistemul propus în cadrul proiectului

Sistemul realizat în acest proiect are ca scop verificarea facială locală, pentru controlul accesului la un vehicul tip autospecială.

Funcționalități cheie:

- Recunoaștere facială offline, printr-o rețea siameză;
- Captură video live și decizie pe loc, fără server extern.
- Transmitere de comenzi către microcontroler prin Bluetooth.
- Interfață grafică pentru adăugarea și gestionarea utilizatorilor.
- Include evitarea obstacolelor cu senzori ultrasonici.
- Nu necesită conexiune la internet pentru nicio etapă.

Avantajele sistemului sunt portabilitatea, independența față de rețele externe și costul scăzut. Limitarea principală este legată de performanța hardware modestă, fiindcă procesarea are loc pe laptop.

Caracteristică	**VeriLook**	**Regula SDK**	**Kairos API**	**Sistem propus**
Detectare liveness	activ + pasiv	activ + pasiv	pasiv	nu momentan
Procesare	on-device	client-server	cloud/local	locală
Potrivire 1:1 / 1:N	ambele	ambele	ambele	doar 1:1
Operare offline	parțial	nu	nu	da
Compatibil embedded	da	limitat	limitat	da
Funcționalitate extinsă	da	da	da	da
Aplicații uzuale	guvernamental, mobil	KYC, frontieră	API cloud	autospecială inteligentă

Tabela 3.1: Comparație între sisteme comerciale și sistemul propus

3.3. Metode

Această secțiune analizează câteva abordări notabile în recunoașterea facială, cu accent pe rețeaua siameză și alternative moderne precum FaceNet și ArcFace. Sunt citate și discutate articole semnificative pentru înțelegerea și poziționarea proiectului propus.

3.3.1. Rețele siameze și învățarea one-shot

Rețelele neuronale siameze sunt folosite pentru a compara două imagini și a decide dacă ele aparțin aceleiași persoane sau clase. Aceste rețele au două ramuri identice care procesează imagini diferite, iar la final generează vectori numerici. Acești vectori (embedding-uri) sunt apoi comparați cu o funcție de distanță, de obicei diferența absolută sau pătratică.

Acest tip de rețea este util mai ales atunci când nu avem multe exemple pentru fiecare clasă. De aceea este des întâlnit în învățarea one-shot. Un exemplu important este lucrarea lui Koch et al. (2015), în care s-a folosit o rețea siameză cu straturi convoluționale pentru a recunoaște caractere din setul Omniglot. Chiar dacă vedea doar o singură imagine dintr-o clasă nouă, modelul putea să recunoască alte imagini asemănătoare [7].

Pentru cazurile în care unele clase sunt slab reprezentate, Guo și Zhang (2017) au propus o metodă care acordă o greutate mai mare acestor clase rare. Ei au introdus o funcție de pierdere specială, numită UP (Underrepresented-classes Promotion), care ajută rețeaua să învețe mai echilibrat [8].

Mai mult, Ding și colaboratorii (2019) au folosit rețele generative pentru a crea imagini noi atunci când avem puține date. Astfel, modelul are mai multe exemple pe care să le folosească în antrenare, chiar dacă ele sunt sintetice. Metoda lor a obținut rezultate foarte bune, cu acurateți de până la 99.80% în unele teste de recunoaștere facială one-shot [9].

În concluzie, rețelele siameze sunt o soluție eficientă în scenarii unde datele sunt limitate. Prin metode de echilibrare sau generare de imagini, performanța lor poate fi semnificativ îmbunătățită, ceea ce le face potrivite pentru aplicații reale, cum este și proiectul prezent.

3.3.2. FaceNet și îmbunătățirea metricii embedding-ului

În recunoașterea facială, o provocare esențială este să găsești o modalitate prin care imaginile diferitelor fețe să fie transformate în reprezentări numerice, astfel încât distanțele dintre ele să reflecte corect asemănările sau deosebirile reale dintre persoane.

FaceNet, propus de Schroff și colaboratorii săi în 2015, abordează această problemă într-un mod diferit de metodele clasice. În loc să clasifice direct imaginile, modelul învață să creeze un spațiu matematic în care fețele aparținând aceleiași persoane sunt cât mai apropiate, iar cele ale altor persoane sunt îndepărtate. Acest lucru se face cu ajutorul unei funcții de pierdere speciale, numită „triplet loss", care folosește trei imagini odată: una de referință, una pozitivă și una negativă.

Un aspect care face FaceNet eficient este felul în care sunt selectate aceste tripleturi. Nu sunt alese la întâmplare, ci sunt căutate acele combinații care ajută rețeaua să învețe diferențele fine dintre fețe. Acest proces, numit „triplet mining", ajută modelul să se concentreze exact pe situațiile în care clasificarea e dificilă.

Rezultatele sunt impresionante: pe setul de date LFW (Labeled Faces în the Wild), FaceNet a obținut o acuratețe de 99,63%, ceea ce l-a făcut rapid una dintre cele mai utilizate abordări în domeniu [10].

Pe partea aplicativă, Arasi (2023) analizează modul în care FaceNet poate fi adaptat pentru utilizări reale. În analiza sa, propune combinarea modelului cu algoritmi de detecție facială (precum MTCNN) și clasificatori suplimentari (de exemplu, SVM) pentru a obține rezultate stabile chiar și în condiții mai puțin ideale – cum ar fi lumină slabă, unghiuri dificile sau expresii faciale neobișnuite [11].

În final, FaceNet a adus o schimbare de perspectivă: în loc să etichetăm direct fiecare față, mai eficient este să învățăm cât de apropiate sunt două imagini. Această idee s-a dovedit foarte practică, dar adaptarea ei în viața reală rămâne dependentă de întregul proces din jurul rețelei, nu doar de modelul în sine.

3.3.3. Pierderi margin-based și standarde moderne: ArcFace

Un progres important în recunoașterea facială a fost dat de introducerea unor funcții de pierdere care pun accent pe distanțarea clară între clasele de persoane. Dintre aceste metode, ArcFace, propus de Deng și colegii săi în 2019, s-a remarcat rapid ca una dintre cele mai eficiente. Principiul din spatele ArcFace este adăugarea unei marje unghiulare între clase, ceea ce ajută modelul să învețe mai bine diferențele dintre persoane care pot arăta foarte asemănător.

Această abordare are și o explicație geometrică intuitivă: în loc să lase clasele prea apropiate într-un spațiu abstract, ArcFace le „împinge" una față de cealaltă pe un cerc unitar, creând un spațiu de reprezentare mai curat. Acuratețe excelentă pe datele de testare peste 99.82% pe setul LFW, este rezultatul. [12].

Un studiu amplu realizat de Srivastava et al. în același an a comparat mai multe tipuri de funcții de pierdere – printre ele, și ArcFace, CosFace, SphereFace. Concluzia lor a fost clară: metodele care folosesc margini unghiulare, precum ArcFace, reușesc să mențină performanțe constante chiar și în condiții dificile – expresii diferite, lumini slabe sau poziții neobișnuite ale feței [13].

Pentru a face sistemul și mai adaptabil, o echipă a propus o variantă numită Sub-center ArcFace. Aici, în loc de un singur centru pentru fiecare clasă, modelul învață mai multe puncte reprezentative. Aceasta ajută în special când o persoană are mai multe „fețe" – de exemplu, purtând ochelari, cu barbă sau zâmbind. Această metodă a crescut

și mai mult acuratețea, ajungând la 99.86% [14].

În concluzie, funcțiile margin-based, în special ArcFace, sunt printre cele mai solide opțiuni în recunoașterea facială modernă. Ele reușesc să combine simplitatea matematică cu rezultate foarte bune în practică.

3.3.4. Alegerea rețelei siameze pentru sistemul propus

Am ales să folosesc o rețea siameză pentru partea de recunoaștere facială din proiect pentru că e una dintre cele mai potrivite soluții atunci când nu ai la dispoziție baze mari de date sau hardware puternic. În comparație cu alte metode mai avansate, precum FaceNet sau ArcFace, care au nevoie de antrenamente complicate și resurse serioase, rețeaua siameză se bazează pe o idee simplă, dar eficientă: în loc să clasifice direct, ea învață să compare două imagini și să decidă dacă e vorba de aceeași persoană sau nu.

Ce este cu adevărat util e că nu trebuie să se reantreneze de fiecare dată când un utilizator nou este adăugat. Practic, este nevoie de o imagine de referință, iar sistemul poate face comparații direct pe baza acelui „model" salvat. E bun pentru scenarii offline, unde nu este internet sau cloud la dispoziție, și mai ales când se dorește un sistem care să funcționeze local.

Avantajele cheie care susțin această alegere:

- **Eficiență la date reduse:** potrivită pentru învățare one-shot sau few-shot, fără nevoia unor seturi extinse per clasă;
- **Rapiditate în procesare:** poate fi rulat local, fără cloud sau GPU, chiar și pe microcontrolere performante;
- **Extensibilitate:** permite înregistrarea de utilizatori noi fără antrenare suplimentară;
- **Robustețe la variații:** demonstrată în lucrări ulterioare ca fiind tolerantă la zgomot, iluminare sau expresii faciale variate [8, 9].

Metodă	**Acuratețe (LFW)**	**Precizie**	**Timp de antrenare**	**Date multe necesare**	**Sensibilitate iluminare**	**Observații**	**Sursă**
CNN clasic	85–90%	Medie	Ridicat	Da	Medie	Nu e scalabil	[15]
Siamese NN	92–96%	Ridicată	Mediu	**Nu**	Mică	Ideal pentru embedded	[7, 8]
FaceNet	99.63%	Foarte ridicată	Mare	Da	Mică	Necesită triplet mining	[10]
ArcFace	99.82%	Foarte ridicată	Foarte mare	Da	Foarte mică	Necesită seturi mari și training intensiv	[12]

Tabela 3.2: Compararea principalelor metode de recunoaștere facială

Capitolul 4. Analiză și fundamentare teoretică

Capitolul acesta are scopul de a prezenta principiile funcționale utilizate care stau la baza aplicației implementate. Sunt descrise metodele, tehnicile și conceptele teoretice folosite. Sistemul realizat este compus din tehnologii hardware, software și machine learning.

4.1. Componente hardware utilizate

În această secțiune am prezentat componentele hardware implicate în realizarea aplicației, alături de principiile tehnice și funcționale care stau la baza utilizării acestora. Am discutat tehnologii de control, protocoale de comunicare, metode de configurare și integrare a componentelor într-un sistem.

Microcontroler STM32F303RE

Placa de dezvoltare NUCLEO-F303RE este produsă de STMicroelectronics și se bazează pe microcontrolerul STM32F303RE, din familia STM32F3. Aceasta oferă un nucleu ARM Cortex-M4, cu frecvență de până la 72 MHz, memorie Flash de 512 KB și RAM de 80 KB. Am ales acest microcontroler datorită capacității de procesare și interfațare. Oferă resursele necesare de memorie și pini configurabili pentru controlul senzorilor, motoarelor și comunicației seriale. Aceasta permite programarea prin USB.[16]

Din punct de vedere al logicii funcționale, microcontrolerul are rolul de a recepționa și a procesa datele provenite de la aplicația software implementată. Funcționează ca o unitate centrală, ce controlează sistemul hardware. Comunică cu perifericele sale prin standarde sau semnale digitale. Deține funcții utile integrate precum comunicație serială, citire de senzori, controlul pinilor, generare de semnal modulație în lățime de impuls (Pulse Width Modulation - PWM) și timere interne. Utilizează în mod special protocolul de comunicație UART (Universal Asynchronous Receiver-Transmitter) pentru transmisia de comenzi, PWM pentru controlul vitezei și timere pentru măsurătorile de precizie. Există un număr mare de pini configurabili GPIO (General Purpose Input/Output) folosiți pentru controlul motoarelor și citirea senzorilor.

- **Biblioteca HAL** (Hardware Abstraction Layer)

  Este un strat software, care simplifică accesul spre perifericele hardware ale microcontrolerului STM32. Abstractizează detaliile nivelului de jos în limbajul C. Elimină accesul direct la registrele perifericelor. Am folosit-o deoarece am considerat că este utilă în dezvoltarea codului pentru că simplifica procesul de configurare și utilizare a perifericelor. Cu ajutorul bibliotecii HAL am inițializat și configurat perifericele GPIO, UART și TIM (Timers). [17]
- **Pinii GPIO (General Purpose Input/Output)**

  Sunt partea fundamentală a resurselor unui microcontroller. Se pot configura individual și reprezintă pinii fizici de pe placa de dezvoltare NUCLEO-F303RE. Pinii se configurează ca ieșiri digitale sau ca intrări digitale. Prin pinii configurați ca ieșiri digitale se trimit semnale către alte componente electronice, cum ar fi LED-

uri (Light Emitting Diode), motoare sau drivere. Pe de altă parte pinii setați ca intrari digitale presupun citirea stărilor senzorilor și semnalelor externe. Fiecare pin este asociat unui port, de exemplu GPIOA sau GPIOB. În cadrul porturilor pinii sunt numerotați în funcție de pozitia lor, exemplu sunt PA1 sau PB3. O altă caracteristică a pinilor este configurarea lor cu sau fără rezistențe de "pull-up/pull-down", precum și viteza de comutare. Caracteristica terminologiei de "pull-up/pull-down" este de a menține valoarea logică 1 stabilă atunci când nu este activ, respectiv valoarea logică 0, în absența unui semnal extern.

Modul Bluetooth HC-05

Modulul Bluetooth HC-05 este un modul de comunicație de tip SPP (Serial Port Profile). Este utilizat pentru comunicarea serială wireless, care operează în banda de frecvențe de 2,4 GHz. Permite implementarea unei conexiuni de tip „master-slave" (dispozitiv principal – dispozitiv secundar). Acesta este des întâlnit în aplicații datorită stabilității conexiunii.[18]

- **Protocolul de comunicație UART**
  Realizează comunicarea cu microcontrolerul. Este interfață serială asincronă cu doar două fire: TX (transmițător) și RX (receptor). Modulul suportă baud rate-uri programabile între 9600 și 460800 bps. În mod implicit, comunicarea AT se face la 38400 bps, fără paritate, cu 8 biți de date și un bit de stop.[18]
- **Comenzile „AT" (eng. Attention Commands)**
  Reprezintă un set de instrucțiuni necesare pentru configurarea modulului HC-05. Parametrii de comunicație ai modulului se pot seta în cadrul acestor instrucțiuni. Comenzile permit setarea numelui dispozitivului, a codului PIN pentru asociere, a vitezei de transmisie (baud rate), a modului de operare (comandă sau date) și a rolului dispozitivului (master sau slave). Aceste caracteristici determină o structură logică și funcțională pentru o conexiune serială fara fir.
- **USB-TTL (eng. Transistor-Transistor Logic)**
  Am utilizat un adaptor USB-TTL compatibil cu nivelul logic de 3.3V pentru configurarea modulului HC-05. Acesta permite comunicarea între calculator și pinii RX/TX ai modulului Bluetooth. Prin acest adaptor se pot transmite comenzile „AT" pentru configurarea cerințelor necesare.

Senzor Ultrasonic HC-SR04

Modulul HC-SR04 este senzor de distanță ce utilizează ultrasunete pentru măsurarea distanței fața de un obstacol aflat în proximitatea acestuia. Principiul de funcționare este unul relativ simplu ce constă în măsurarea intervalului de timp între un impuls ultrasonic emis și recepția ecoului reflectat de obstacol. Utilizează două semnale digitale "TRIG" și "ECHO". Fiecare pin are o funcționalitate diferită. Pinul TRIG este responsabil cu inițierea măsurătorii: acestea primește un impuls scurt, de minimum 10 µs și ulterior emite un semnal ultrasonic. După această undă emisă, senzorul va activa pinul ECHO constant, cu valoarea HIGH (valoare logică 1), până în momentul în care semnalul propagat se întoarce spre senzor. Timpul dat de acest interval este folosit de microcontrolerul STM32 pentru calcularea distanței. Senzorul necesită un timer intern pentru cronometrarea semnalului generat de pinul ECHO. În figura (4.1) este prezentat principiul descris anterior.

Funcționează la o tensiune de alimentare de 5V și are un consum de curent redus, de aproximativ 15mA. L-am utilizat în acest proiect pentru capacitatea sa în oprirea automată a robotului prin generarea de semnale, în cazul detectării unui obstacol aflat în exteriorul acestuia.

$$\text{Distanța (cm)} = \frac{\text{Timpul măsurat } (\mu s) \times 0{,}034}{2} \tag{4.1}$$

Constanta 0,034 reprezintă viteza sunetului în aer, exprimată în cm/µs. Timpul măsurat este exact durata în care semnalul ECHO rămâne la nivel logic HIGH. [19]

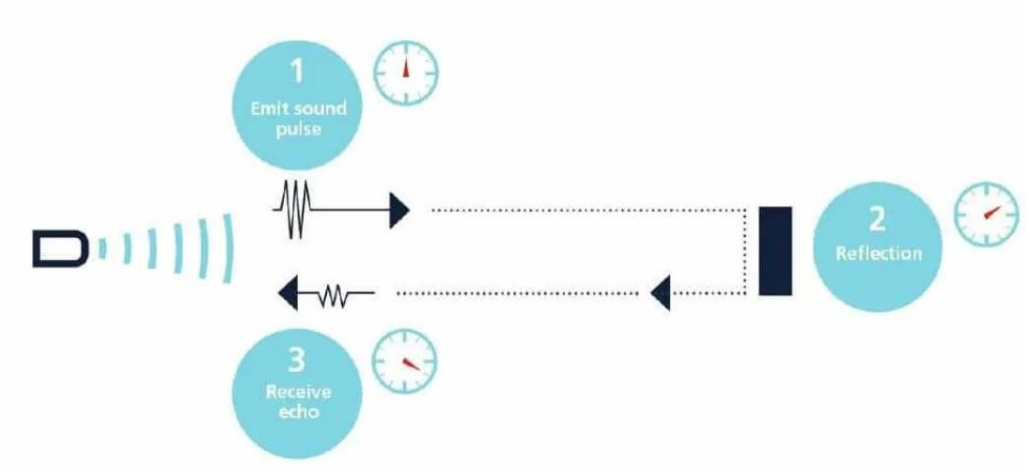


Figura 4.1: Principiul de funcționare al senzorului ultrasonic HC-SR04 [20]

Motoare DC

Motoarele de curent continuu (eng. DC) sunt componente electromagnetice care transformă energia electrică în energie mecanică. Funcționează prin aplicarea unei tensiuni electrice la bornele motorului și generează un câmp magnetic ce produce rotația axului. Se poate defini direcția de rotație prin modificarea polarității tensiunii ce va rezulta controlul mișcării înainte și înapoi a roții atașate motorului.

- **Semnalul PWM (Pulse Width Modulation)**

  Această tehnică se folosește pentru controlul vitezei. Este un semnal digital ajustabil, definit de către microcontroler. Valorile semnalului digital comută între valoarea logica 0 și valoarea logica 1 (OFF și ON). Cantitatea de energie este reglabila prin modificarea duratei semnalului ON, numit ciclu de funcționare - duty cycle. Ciclul de funcționare este exprimat în procente și reprezintă media de energie electrică livrată, conform ecuației (4.2):

$$D[\%] = \frac{CCR}{ARR} \cdot 100 \tag{4.2}$$

  unde:
  - CCR – valoarea registrului de comparatie (eng. capture/compare register);
  - ARR – valoarea maximă a timerului (eng. auto-reload register).

  În figura 4.2, se poate observa semnalul PWM cum este definit prin alternarea a două niveluri de tensiune, cu o frecvență constantă și o durată de impuls variabilă.

Controlul motoarelor DC acționate de semnalul PWM necesită utilizarea unui timer hardware. Timerul generează semnalul PWM cu frecvență și ciclu de lucru configurabil. Microcontrolerul STM32F303RE dispune de mai multe timere (ex. TIM1, TIM2) necesari în scopul producerii semnalelor PWM pe canale dedicate.
Un factor important în generarea semnalului PWM este frecvența semnalului determinată de repetiția impulsurilor pe secundă, conform ecuației (4.3).

$$f_{PWM} = \frac{f_{CLK}}{(PSC + 1) \cdot (ARR + 1)} \tag{4.3}$$

unde:
- f_{CLK} – frecvența de tact a timerului
- PSC – valoarea prescalerului (prescaler)
- ARR – valoarea registrului de auto-reîncărcare (auto-reload register).

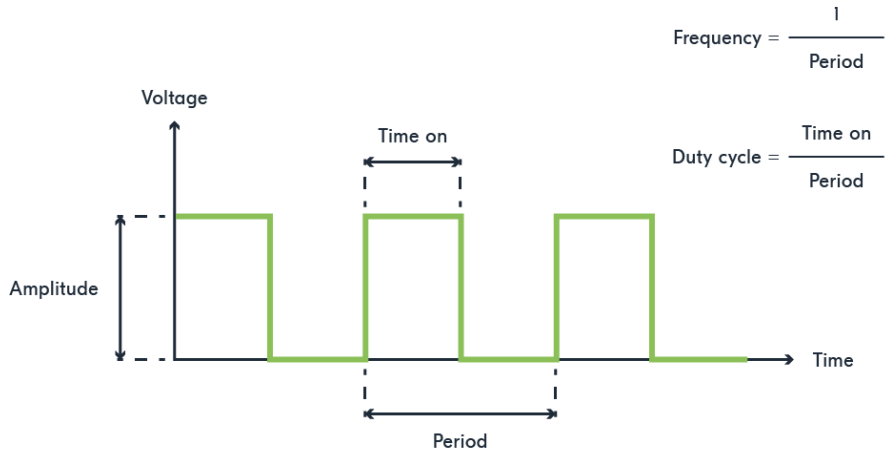


Figura 4.2: Exemplu de semnal PWM. Imagine preluată de pe Nordic Semiconductor Academy [21].

Modul driver L298N - Punte H dublă

- **Puntea H (eng. H-bridge)**
  Este un circuit electronic simplu și are o propritate necesara în construirea robotului masina. Are capacitatea de a controla direcția de rotație înainte sau înapoi, a unui motor DC.

  Din punct de vedere structural, puntea H, este compusă din 4 comutatoare de tip tranzistori. Sunt pozitionate sub forma literei H, de unde și proveniența numelui. În figura 4.3 este reprezentarea conceptuală. Circuitul reprezentat în schema logica este un circuit compus din 4 tranzistori Q1-Q4 ce sunt conectați într-o formă simetrica în jurul motorului DC. Tranzistorii sunt conectați în perechi obligatorii opuse, altfel se va produce un scurtcircuit între VCC (tensiunea de alimentare) și GND (masa, valoarea 0V). Perechile opuse sunt Q1-Q4 și Q2-Q3. Sensul direcției

este determinat de perechea activată (ex. Q1-Q4 înainte sau Q2-Q3 înapoi) [22]. Semnalul PWM generat de microcontroler este preluat de puntea H și trimis spre bornele motorului.

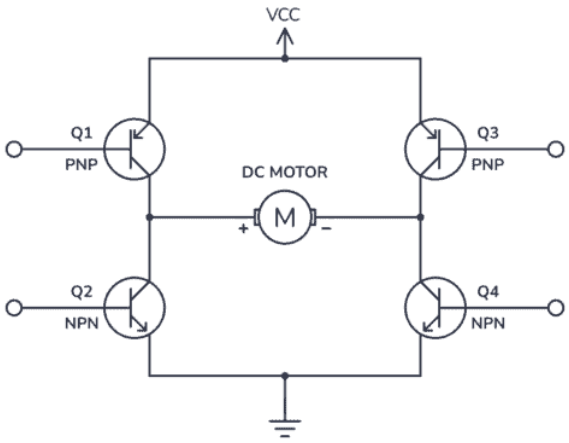


Figura 4.3: Schema logică a unei punți H. [22].

- **Modulul L298N**

  Modulul are două circuite integrate de tipul punte H, fiecare este independentă. Funcționează la o tensiune de alimentare de 5V până la 35V și o tensiune logică de 5V. Conține două perechi de ieșiri OUT1, OUT2 și OUT3, OUT4. Are pini de intrare separați pentru două semnale de input ce sunt direcționate spre motoare. Sunt prevăzuti și doi pini ENA (eng. enable motor A) și ENB ce dicteaza activarea motoarelor din componența robotului Vezi figura4.4. [23]

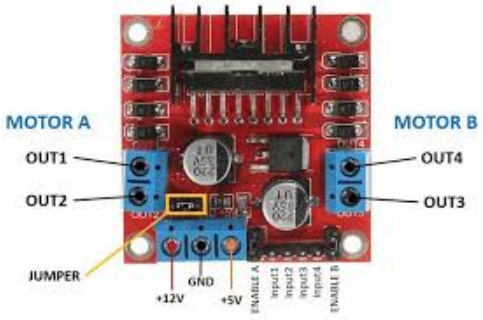


Figura 4.4: Modul driver L298N
[24]

4.2. Parte software

Aplicația am realizat-o ca o interfață grafică destinată interacțiunii cu utilizatorul pentru gestionarea accesului. Are un sistem modular cu mai multe ferestre. Aplicația este organizată sub forma unei aplicații cu mai multe ferestre (eng. multi-view). Soluția am dezvoltat-o în limbajul Python și am utilizat biblioteca PyQt5 pentru realizarea interfeței grafice.

Componentele software și cele mai importante biblioteci pe care le-am utilizat sunt:

- PyQt5: pentru construirea interfeței grafice GUI (Graphical User Interface) , cu ecrane, butoane, zone de încărcare imagini, etc.
- OpenCV: pentru accesarea camerei web și prelucrarea imaginilor capturate (afișare, redimensionare, conversie RGB).
- MTCNN (Multi-task Cascaded Convolutional Networks): pentru detecția facială din imagini.
- TensorFlow / Keras: pentru încărcarea modelului de recunoaștere facială descris în lucrare.
- NumPy: pentru operații matematice pe datele imaginii și prelucrare vectorială.
- PySerial: pentru comunicarea serială cu un port COM.
- QProcess: pentru rularea unor scripturi externe (ex. reantrenare/fine-tuning).
- QFileDialog / QScrollArea / QLabel: pentru încărcare fișiere, afișare imagini și organizare grafică.

Limbaj de programare și paradigmă utilizată

Pentru dezvoltarea aplicației am utilizat limbajul Python deoarece un limbaj de programare interpretat. Este utilizat frecvent pentru analiza datelor, automatizări și, în contextul ales de mine, dezvoltare de aplicații și inteligență artificială. Python este accesibil, ceea ce m-a ajutat în special la întreținerea codului. În proiectul meu, Python este folosit pentru a controla pașii algoritmici, pentru procesarea imaginilor și pentru comunicarea cu alte componente hardware, cum ar fi microcontrolerul sau modulul Bluetooth. Limbajul permite lucrul atât cu clase și obiecte precum programarea orientată pe obiect, dar și cu funcții simple pentru programarea procedurală. Am utilizat Python datorită suportului pentru lucrul cu fișiere, operații matematice și comunicație serială. În acest fel, în aplicația rezultată am integrat diverse funcții fără a fi nevoie de cod complex.

Interfață grafică: PyQt5

Pentru realizarea interfeței grafice am folosit biblioteca PyQt5. Aceasta este o extensie a limbajului Python care permite folosirea funcțiilor oferite de frameworkul Qt. Qt este un framework folosit pentru dezvoltarea de interfețe grafice în aplicații scrise în C++. Am considerat că PyQt5 este util deoarece oferă acces la folosirea aceleași funcții și în Python. Astfel, am putut crea ferestre, butoane, etichete și alte elemente direct cu limbajul Python. PyQt5 permite conectarea acestor elemente la funcții din aplicație și face posibilă interacțiunea cu utilizatorul. În proiectul meu, am folosit această bibliotecă pentru a crea o interfață prin care utilizatorul poate controla aplicația fără să folosească linia de comandă.

Captură video și procesare imagini: OpenCV

În contextul aplicației mele am folosit biblioteca OpenCV pentru a lucra direct cu camera laptopului. Cu ajutorul ei am reușit să accesez imaginea live, cadru cu cadru, și să o trimit mai departe către componentele de procesare. Imaginea este preluată în format BGR (standard OpenCV), iar pentru afisare sau pentru recunoaștere facială, am convertit-o în RGB. Apoi, cadrele le-am redimensionat astfel încât să se potrivească cu dimensiunile interfeței grafice create cu tehnologia PyQt5.

Pe lângă captură, OpenCV m-a ajutat să fac și pre-procesarea imaginilor. Am aplicat redimensionare, normalizare și uneori și conversii în alb-negru sau detecții de fețe (pentru extragerea porțiunii de interes). În combinație cu modelul de recunoaștere

facială, am folosit aceste imagini pentru a verifica dacă persoana este autorizată. Dacă da, se trimite un semnal mai departe prin Bluetooth către microcontroler.

OpenCV este o bibliotecă folosită des în domeniul procesării de imagini. Dispune de multe funcții pentru lucrul cu imagini, cum ar fi detecția de obiecte, contururi, filtre, transformări geometrice și altele. În cazul proiectului meu, a fost utilă deoarece a conectat partea vizuală cu partea de control și decizie.

Comunicare serială: PySerial

În aplicația mea, am folosit comunicarea serială ca să pot trimite comenzi simple către un dispozitiv extern. Totul se face printr-un port COM, în funcție de cum este conectat modulul Bluetooth. Am folosit o bibliotecă din Python, pyserial, care îmi permite să deschid o conexiune și să scriu direct în acel port. Dacă portul nu e conectat sau e greșit, aplicația știe și anunță utilizatorul.

Ca semnale trimit litere precum „w" sau „a", ele sunt înțelese de dispozitiv ca semnale de control. Dispozitivul reacționează la caracterele trimise de mine prin port. Este un mod simplu de comunicare între aplicație și sistemul fizic, de aceea am alse sa il integrez ca parte principala de comunicare între cele două componente.

Execuție scripturi externe: QProcess

În aplicație am avut nevoie să rulez un script separat, care se ocupă de reantrenarea modelului. Nu am dorit ca aplicația să se blocheze cât timp se face acest proces, așa că am folosit QProcess. Cu el am lansat un script Python extern direct din aplicație, fără să închid fereastra principală.

Procesul pornește într-un fel „în fundal", dar totuși văd ce se întâmplă, pentru că outputul din script apare într-o fereastră din aplicație. Acolo se încarcă toate mesajele, rând cu rând, exact cum ar apărea în terminal. Dacă apare vreo eroare, se afișează și ea, colorată în roșu.

După ce se termină rularea scriptului, utilizatorul primește un mesaj că finetuningul s-a terminat. În concluzie, cu QProcess pot lega aplicația de un proces extern și totul se întâmplă fluid din perspectiva unui utilizator.

Tehnologii complementare

NumPy: Am folosit această bibliotecă pentru utilitatea sa în calculul numeric. În implementarea actuală, am folosit-o pentru manipularea matricelor de pixeli, reprezentarea imaginilor stocate sub formă de array-uri. O altă funcționalitate a acestei biblioteci este efectuarea unor operații statistice, cum ar fi calculul mediei scorurilor de similaritate rezultate din modelul de recunoaștere facială. Am ales-o deoarece în bibliotecă sunt incluse funcții compatibile cu procesarea imaginilor sau procesarea seturilor mari de date.

os / shutil: Pentru gestionarea fișierelor utilizez aceste biblioteci fundamentale în lucrul cu sistemul de operare. Sunt folosite pentru a accesa, crea, sau șterge directoare destinate organizării imaginilor. Totodată, prin intermediul modulului `shutil`, am realizat operațiuni precum mutarea sau copierea fișierelor între diferite locații din sistem. Aceste funcționalități ajută atât pentru salvarea imaginilor prelucrate, cât și pentru pregătirea seturilor de date utilizate în antrenarea și testarea modelului.

Structuri Qt (QVBoxLayout, QGridLayout): Am folosit aceste structuri pentru a așeza în mod organizat elementele grafice în fereastra aplicației. Cu ajutorul layout-urilor precizate, am aranjat butoanele, etichetele și zonele de afișare într-o manieră clară și ușor de urmărit.

QInputDialog / QMessageBox:Interacțiunile între aplicație și utilizator le-am conturat prin aceste două componente. Pentru introducerea unor date de la tastatură am folosit QInputDialog, iar QMessageBox pentru afișarea mesajelor de confirmare sau notificare.

4.3. Machine Learning

TensorFlow/Keras – cadrul de dezvoltare utilizat

În această secțiune am prezentat framework-ul TensorFlow/Keras. Acest cadru l-am utilizat pentru implementarea rețelei neuronale dezvoltată în acest proiect. Keras este o interfață de nivel înalt, integrată în TensorFlow. Face parte din categoria bibliotecilor de învățare profundă (eng. deep learning).

Principiu de funcționare: TensorFlow funcționează pe baza unor grafuri simbolice computaționale care permit efectuarea operațiilor matematice pentru rețelele neuronale, atât pe CPU, cât și pe GPU. Un graf simbolic reprezintă o structură în care nodurile corespund operațiilor matematice, iar muchiile indică fluxul de date între aceste operații. Timpul de execuție este optimizat prin intermediul acestor grafuri. Keras, integrat în TensorFlow, oferă o interfață structurată pentru definirea arhitecturii rețelelor. Datorită acestor capacități, am ales să dezvolt acestă aplicație utilizând TensorFlow.

În aplicația dezvoltată, am utilizat TensorFlow/Keras pentru:

- Construirea unei rețele convoluționale (CNN) specializate în extragerea vectorilor de trăsături faciale.
- Configurarea arhitecturii rețelei Siamese.
- Implementarea stratului (eng. layer) de diferență L1, urmat de funcția de activare sigmoid.
- Compilarea modelului cu funcția de pierdere de tip entropie binară (Binary Cross-Entropy) și algoritmul de optimizare Adam;
- Antrenarea modelului, care folosește date procesate și stocate local;
- Salvarea modelului antrenat într-un format compatibil cu utilizarea ulterioară în aplicația finală.

Valoare teoretică și aplicabilitate: Utilizarea framework-ului TensorFlow permite definirea unei arhitecturi personalizate, bazată pe rețele neuronale convoluțional. Se folosesc structuri specifice învățării pe perechi (eng. pair-based learning).În straturile utilizate sunt incluse componente standard precum Conv2D, MaxPooling2D, Flatten și Dense. Straturile și componentele le-am adaptat cerințelor unei rețele pentru verificare facială.

TensorFlow oferă suport pentru:

- Persistența modelului – salvarea și încărcarea modelului antrenat fără pierderea performanței.
- Integrarea cu biblioteci externe precum OpenCV (pentru procesarea imaginilor) și NumPy (pentru manipularea numerică a datelor).
- Compatibilitate cu formatul standard de salvare a modelelor (.h5), este necesar

pentru aplicarea modelului în etapa dezvoltării.

One-shot Learning – Generalizare cu date minime

În scenariile clasice de învățare automată, antrenarea unui model presupune sute sau mii de exemple pentru a învăța să recunoască o clasă. În acest caz, în verificarea facială, pentru adăugarea unui utilizator nou sunt necesare doar câteva imagini. Pentru rezolvarea acestei probleme am utilizat paradigma de „One-Shot learning" - Învățare bazată pe un singur exemplu.

Definiția și principiul funcțional: În tehnica One-shot learning – modelul trebuie să recunoască o clasă nouă văzând un singur exemplu. Nu se bazează pe memorarea claselor, ci pe învățarea unei funcții de similaritate. Funcția respectivă permite compararea între un exemplu necunoscut și exemplele de referință.

Un astfel de comportament se obține prin:

- Învățarea unui spațiu de embedding în care distanțele reflectă similaritatea imaginilor. Distanțele sunt reprezentate prin vectori numerici.
- Antrenarea modelului este constituită din perechi pozitive/negative, nu pe etichete de clasă.

Valoare teoretică și aplicabilitate: Rețelele de tip Siamese sunt modele clasice pentru one-shot learning. Modelul învață să compare imagini, nu să le clasifice direct. Poate sa decidă dacă două imagini reprezintă aceeași persoană și în cazul în care acea persoană nu a fost văzută în timpul antrenării. După antrenare modelul se poate extinde la orice utilizator nou. Dacă s-au adăugat noi utilizatori, prin această tehnică, nu mai este necesar să se reia procesul de antrenare. Metoda aceasta îmi permite să adaug numai imagini de referință definite conceptual „ancoră". Această metodă se potrivește aplicațiilor de control al accesului și recunoaștere biometrică, în special cu resurse și date limitate.

Siamese Neural Network – Model de comparare a similarității

În centrul aplicației propuse am integrat un model neuronal numit Siamese Neural Network (SNN). L-am utilizat pentru verificarea identității unei persoane pe baza similarității dintre imagini cu alte persoane. Din punct de vedere tehnic, rețeaua are o arhitectură dual. Această arhitectură este proiectată să compare două obiecte și să determine dacă acestea aparțin aceleiași clase.

Modelul este format din două rețele neuronale convoluționale identice (denumite „twin networks"). Modelul procesează în paralel două imagini de intrare. Fiecare rețea extrage un vector de caracteristici (embedding) care reprezintă trăsăturile imaginii. Acești doi vectori apoi i-am comparat printr-o funcție de distanță (ex. L1, L2) și rezultatul l-am interpretat drept scor de similaritate. Am ales distanța L1 pentru compararea embeddingurilor.

Principiul funcțional:

1. Intrare: două imagini, o imagine de referință (ancoră) și una de verificat.
2. Procesare paralelă: ambele imagini trec prin rețele CNN identice, cu parametri partajați.
3. Extracție de trăsături: fiecare rețea produce un vector numeric ce îmbină caracteristicile feței.

4. Comparare: diferența dintre vectori este introdusă într-un strat final care produce o valoare de similaritate între 0 și 1.

Rețelele Siamese sunt optimizate pentru învățarea relațiilor dintre perechi de date. Nu există termenul de clasificare. Rețeaua învăța să generalizeze similaritatea și la persoane neîntâlnite în setul de antrenare.

Am ales aceată rețea deoarece am considerat că se potriveste cu sarcina de verificare biometrică. În verificarea bometrică nu este necesară etichetarea pe categorii. Scopul final este de a confirma identitatea.

CNN –Rețea Neuronală Convoluțională

În opinia mea, un element esențial în orice sistem de recunoaștere facială este rețeaua neuronală convoluțională (CNN). Am ales acest tip de rețea deoarece este potrivit pentru procesarea imaginilor. Este indicată să se utilizeze și în extragerea automată a trăsăturilor vizuale. CNN-urile sunt utilizate în domeniul viziunii artificiale pentru sarcini precum clasificare, detecție sau segmentare. În cazul proiectului meu, aceste rețele constituie codificarea informației extrasă din trăsaturile feței.

Mod de funcționare: Modelul CNN funcționează prin aplicarea succesivă a unor filtre convoluționale peste imagine, pentru a detecta în primă fază elemente simple (cum ar fi margini sau colțuri). Ulterior se detectează forme mai complexe cum ar fi ochii, nasul sau conturul feței.

După aceste straturi convoluționale, am aplicat:

1. Funcția de activare ReLU, pentru a introduce non-linearitate;
2. Straturi de pooling, specific am ales max pooling, pentru a reduce dimensiunea datelor și a păstra trăsăturile de bază
3. Straturi dense, care proiectează rezultatul final într-un spațiu de embedding, sub formă de vector numeric.

În ultimul pas, imaginea brută este transformată într-un vector de trăsături ce poate fi comparat în cadrul rețelei Siamese. Această abordare permite identificarea diferențelor între fețe fără a fi necesară extragerea manuală a trăsăturilor. Modelul învață singur, ce caracteristici sunt importante pentru diferențiere. După aceste straturi convoluționale, am aplicat trei tipuri de componente esențiale:

1. **Funcția de activare ReLU**
   Am aplicat ReLU (Rectified Linear Unit) după fiecare strat convoluțional, pentru a introduce non-linearitate în rețea. Aceasta păstrează doar valorile pozitive și înlocuiește valorile negative cu zero. Introduce non-linearitate în rețea fără să afecteze viteza de calcul. Am ales să includ această tehnică deoarece este un standard în arhitecturile CNN pentru a accelera antrenarea și reduce problemele de tip „vanishing gradient". Probleme care apar când valorile gradientului devin foarte mici în timpul antrenării. Valorile foarte mici împiedică rețeaua să învețe eficient în straturile de la început.
2. **Straturi de pooling (max pooling)**
   După straturile convoluționale am aplicat max pooling, pentru a reduce dimensiunea matricei de activare. Acest strat menține informațiile importante și reduce numărul total de parametri. în final rețeaua o să fie mai eficientă.
3. **Straturi dense (Fully Connected)**

La finalul rețelei CNN am adăugat un strat dens (fully connected) care transformă datele în vectori numerici (embedding). Acești vectori sunt folosiți ulterior în rețeaua Siamese pentru compararea similarității între două fețe. Abordare permite identificarea diferențelor între fețe fără a fie necesară extragerea manuală a trăsăturilor. Modelul învață singur ce caracteristici sunt importante pentru diferențiere.

Justificarea alegerii: Am optat pentru CNN datorită capacității sale bune de generalizare. Este eficient din punct de vedere computațional, mai ales datorită partajării greutăților în spațiu (convoluția fiind locală), aspect esențial în aplicațiile embedded sau cu resurse limitate. Se integrează ușor în cadrul platformelor de deep learning moderne, precum TensorFlow/Keras, pe care le-am folosit și eu în acest proiect.

Regularizare – L2 și Dropout

În arhitectura rețelei convoluționale am inclus două metode de regularizare, pentru a preveni supraînvățarea (eng. overfitting) și pentru a crește capacitatea modelului de a generaliza. Prima este regularizarea de tip L2 și este aplicată pe toate straturile convoluționale. Metoda penalizează greutățile prea mari și încurajează un model mai simplu. A două metodă este Dropout, cu o rată de 0.4, aplicată înainte de stratul final. Aceasta dezactivează aleatoriu o parte din neuroni în timpul antrenării, astfel încât modelul să nu se bazeze prea mult pe o anumită combinație de trăsături.

Am ales aceste metode pentru că s-au potrivit bine cu rețeaua pe care am construit-o. Fiind vorba de date reale, cum am precizat, cu multe variații între imagini, modelul trebuia să nu învețe „pe de rost" anumite trăsături. Regularizarea a ajutat ca antrenarea să fie mai stabilă, fără să apară diferențe mari între setul de antrenare și cel de validare.

Funcția de pierdere – Binary Cross-Entropy vs. Contrastive Loss

Pentru antrenarea rețelei neuronale a trebuit să aleg o funcție de pierdere care să reflecte corect scopul aplicației mele: verificarea identității unei persoane pe baza similarității dintre două imagini. În cadrul rețelelor Siamese sunt utilizate două funcții de pierdere: „Contrastive Loss" și „Binary Cross-Entropy" (BCE). După testare și documentare, am optat pentru varianta BCE.

Contrastive Loss este o funcție specifică pentru învățarea metrică (eng. metric learning), folosită pentru a antrena rețelele să „apropie" în spațiul de embedding vectorii corespunzători imaginilor similare și să „depărteze" cei ai imaginilor diferite.

Funcția 4.4 poate da rezultate bune în modele axate exclusiv pe distanță, însă în testele mele s-a dovedit mai instabilă în prezența variațiilor de calitate sau lumină din datele reale. De asemenea, necesită o atenție specială pentru alegerea marjei și interpretarea rezultatului.

$$L = (1 - y) \cdot \frac{1}{2} \cdot D^2 + y \cdot \frac{1}{2} \cdot (\max(0, m - D))^2 \tag{4.4}$$

unde:

- $y \in \{0, 1\}$ reprezintă eticheta (0 pentru perechi similare, 1 pentru perechi diferite),
- D este distanța dintre vectorii de embedding ai imaginilor
- m este o marjă definită (margin) care controlează cât de depărtate trebuie să fie perechile diferite.

Binary Cross-Entropy (BCE) este o funcție de pierdere clasică, folosită în clasifi-

carea binară. Ea evaluează cât de apropiată este predicția modelului (după sigmoid) față de eticheta reală (0 sau 1).

În opinia mea, funcția BCE dată de formula 4.5, se portivește în contextul aplicației mele deoarece:

- Este stabilă în fața zgomotului din date și a variațiilor de calitate a imaginilor.
- permite aplicarea unui prag (eng. threshold) în etapa de rulare și evaluare a modelului.
- este compatibilă cu funcția de activare sigmoid din stratul final.

$$L = -\left[y \cdot \log(p) + (1 - y) \cdot \log(1 - p)\right] \tag{4.5}$$

- $y \in \{0, 1\}$ reprezintă eticheta reală
- $p \in [0, 1]$ este probabilitatea prezisă de model, obținută după aplicarea funcției sigmoid,
- L este valoarea pierderii (eng. loss) pentru singur exemplu din setul de date

Funcția de pierdere Binary Cross-Entropy (BCE) am utilizat-o pentru a evalua acuratețea modelului în cadrul fiecărei serii de date (eng. batch). Alegerea acestei funcții s-a demonstrat că este practică în proiect. Contribuie la obținerea performanțelor în recunoașterea identității pe imagini neștiute anterior.

Optimizatorul Adam

În timpul antrenării rețelei neuronale, algoritmul de optimizare are un rol important. Am ales să folosesc Adam (eng. Adaptive Moment Estimation), deoarece vreau să folosesc un optimizator stabil.

Cred că este important ca modelul să ofere rezultate constante, chiar și atunci când datele conțin zgomot, iluminare variabilă sau imagini de calitate slabă. Am căutat o variantă care să nu genereze oscilații mari în pierdere sau acurateță. Adam s-a dovedit potrivit pentru seturi de date reale, nesintetice, specific datelor mele.

Nu necesită reglaje complicate. Folosește hiperparametri standard, funcționali fără ajustări. Nu e nevoie de căutare manuală pentru rata de învățare, iar configurarea este simplă. În cazul meu, am făcut o singură modificare: am setat rata de învățare (eng. learning rate) la $1 \cdot 10^{-4}$, o valoare suficient de mică pentru a asigura stabilitate. Restul parametrilor au rămas la valorile implicite. Această configurare minimă a fost suficientă pentru ca modelul să învețe corect, fără instabilități.

Calculul scorului de similaritate – Sigmoid și Distanța L1

- **Funcția de activare Sigmoid** :

  În stratul final al rețelei, am folosit o funcție de activare de tip Sigmoid. Alegerea acestei funcții a fost necesară pentru a transforma scorul calculat de rețea într-o valoare între 0 și 1, interpretabilă ca probabilitate. Am avut nevoie de o ieșire să reflecte gradul de similaritate dintre cele două imagini, astfel încât să pot să aplic un prag (eng. threshold) clar pentru luarea deciziei. Sigmoidul este o funcție simplă și eficientă în sarcini de clasificare binară, iar în cazul meu, a funcționat bine în combinație cu funcția de pierdere Binary Cross-Entropy. Scorurile obținute după activarea Sigmoid au fost ușor de interpretat și au permis o evaluare clară în timpul testării.

- **Distanța L1**:

Pentru a compara vectorii extrași de cele două rețele, am folosit diferența absolută între ele, cunoscută ca distanța L1. Această alegere a venit din nevoia de a obține o măsură simplă a diferențelor dintre trăsăturile faciale. Distanța L1 calculează cât de „departe" sunt cele două reprezentări numerice. Am ales L1 în defavoarea distanței L2 să evit valorile extreme. În contextul rețelei Siamese, această distanță mi-a permis să evidențiez diferențele relevante dintre imagini într-un mod stabil, care s-a integrat cu stratul de ieșire. A fost un pas important în construcția unui scor de similaritate cât mai real.

Optimizare prin reantrenare - Fine-Tuning

În acest sistemul de recunoaștere facială, am introdus conceptul de fine-tuning. Scopul lui este de a actualiza modelul deja antrenat atunci când se adăugă utilizatori noi. A nu se reantrena modelul de la început, acesta învață doar din noile imaginile. În contextul proiectului, modelul păstrează tot ce a învățat anterior dar și noile date.

Pentru fiecare utilizator nou, când o persoana este nouă în sistem imaginile sunt adăugate în setul deja existent, iar modelul este reantrenat pe toate perechile generate.

Din punct de vedere logic, sistemul conține două tipuri de imagini: cele ale utilizatorilor înregistrați și cele negative, care nu aparțin niciunui utilizator cunoscut. Acestea sunt combinate pentru a forma perechi care sunt folosite la antrenare.

Am ales această metodă pentru că, pe măsură ce se adaugă persoane noi, vreau ca modelul să fie capabil să le recunoască corect. Fără fine-tuning, există riscul ca sistemul să greșească atunci când întâlnește fețe necunoscute, pentru că nu a fost antrenat cu ele. După analiză, am ajuns la concluzia că integrarea fine-tuning-ului contribuie la creșterea preciziei sistemului.

4.4. Prelucrarea Datelor

Procesarea imaginilor este o etapă importantă într-un sistem de recunoastere facială. Pregătirea imaginilor brute este pasul precedent antrenării modelului. Ca flux logic de pregătire a datelor am inclus detecția feței, decuparea, normalizarea, augmentarea imaginilor și generarea perechilor pozitive și negative pentru învățare.

În vederea antrenării rețelei siameze pentru recunoașterea facială, am realizat o serie de pași pentru preprocesarea și structurarea datelor. Am folosit două categorii de imagini: persoane „cunoscute" (pozitive) și persoane „necunoscute" (negative). Astfel modelul învață sa diferențieze fețele asemănătoare de fețele complet diferite.

Selecția seturilor de date

Am definit clasele pozitive și negative prin cominarea mai multor imagini din seturi deja existente.

Pentru clasa pozitivă, am intercalat imagini din două seturi deja existente. Selecția a fost de aproximativ 50 de persoane, fiecare cu minim 90 de imagini diferite. Am alternat cele două surse pentru a diversifica datele. Seturile de imagini utilizate pentru clase le-am preluate din surse publice disponibile pe platforma Kaggle[1]. Kaggle mi-a oferit o varietate de dataseturi pentru dezvoltare în domeniul învățării automate.

- **VGGFace2**[2] – un set de date divers cu imaginile unor persoane diferite în condiții variate.

[1]Kaggle se poate accesa pe site-ul oficial: `https://www.kaggle.com/`

[2]VGGFace2: `https://www.kaggle.com/datasets/hearfool/vggface2/data`

- **Celebrity Face Image Dataset**[3] – set cu fotografii clare ale unor persoanelor publice.

Pentru clasa negativă, am folosit aproximativ 2000 de imagini cu persoane necunoscute din folderul `real faces` al datasetului **Human Faces Dataset**[4], disponibil pe Kaggle.

Am verificat manual ca persoanele din setul pozitiv să nu se regăsească și în cel negativ. M-am asigurat că fiecare imagine utilizată nu conține mai multe fețe. Am respectat unicitatea fotografiilor, în scopul unei antrenări curate.

Detectarea și decuparea fețelor

Pentru a standardiza dimensiunile și a elimina elementele din fundal, am aplicat un proces de detectare automată a feței cu ajutorul algoritmului **MTCNN**. Fiecare imagine a fost trecută printr-o etapă de decupare (eng. crop) a regiunii feței, urmată de redimensionare la 105×105 pixeli. Această etapă am realizat-o pentru a pregăti corect intrările rețelei neuronale.

Algoritmul MTCNN

Pentru a extrag automat regiunea feței din imagini, am utilizat algoritmul MTCNN (Multi-task Cascaded Convolutional Networks). Acesta extrage fața din imagine și returnează coordonatele dreptunghiului ce o încadrează. Fața detectată urmează să fie decupată și redimensionată uniform la 105x105 pixeli. Aceste dimensiuni sunt dimensiunile standard pentru rețeaua neuronală.

Detectarea MTCNN asigură procesul corect de detecție a feței din imagine. Poziția sau dimensiunea feței în cadrul imaginii nu este relevantă deoarece algoritmul ales menține toate imaginile rezultate la aceleasi dimensiuni și poziționări. Această etapă are rolul de a uniformiza datele de intrare, și confirma învățarea consistentă a modelului.

MTCNN funcționează pe baza unei arhitecturi formate din trei rețele convoluționale care operează în cascadă:

- **P-Net (Proposal Network)** – detectează regiuni candidate pentru fețe;
- **R-Net (Refine Network)** – elimină propunerile incorecte și rafinează locația regiunilor;
- **O-Net (Output Network)** – finalizează localizarea și returnează coordonatele feței și ale punctelor de interes (ochi, nas, gură).

MTCNN localizează dreptunghiul ce încadrează fața, ci poate detecta și trăsături faciale (eng. landmarks). În cazul proiectului de față, am utilizat doar coordonatele pentru delimitarea feței (*bounding box*).

Alegerea algoritmului MTCNN este justificată prin mai multe avantaje pe care le-am observate în procesul de prelucrare a datelor. În primul rând, acesta oferă o detecție precisă a fețelor chiar și în imagini variate din punct de vedere al luminii, orientării sau expresiei faciale. De asemenea, am remarcat o stabilitate în identificarea corectă a feței principale, fără generarea de fals pozitive.

[3]Celebrity Face Image Dataset: se poate descărca de la adresa `https://www.kaggle.com/datasets/hearfool/vggface2/data`

[4]Human Faces Dataset: se poate descărca de la adresa `https://www.kaggle.com/datasets/hearfool/vggface2/data`

Un alt criteriu important a fost viteza cu care algoritmul procesează imaginile. În plus, rezultatele detecției sunt consistente și redimensionează uniform regiunea feței la dimensiunea necesară rețelei mele, asigurând astfel o intrare standardizată pentru întregul model.

Preprocesare și augmentare

Pentru a evita suprainvățarea, am aplicat o serie de transformări asupra imaginilor:

- normalizare în intervalul $[0, 1]$
- rotiri și inversări orizontale aleatorii
- variații ușoare de luminozitate, contrast, saturație și nuanță
- zgomot JPEG controlat pentru realism

Aceste augmentări au fost aplicate doar pe setul de antrenare, pentru a nu altera datele de validare.

Pentru a pregăti corect imaginile înainte de antrenare, fiecare imagine a fost transformată într-un format numeric adaptat cerințelor rețelei neuronale. După detecția și decuparea feței, imaginea este convertită într-un tensor tridimensional de dimensiune $105 \times 105 \times 3$, unde fiecare pixel are valori normalizate în intervalul $[0, 1]$. Această normalizare a fost necesară pentru a asigura stabilitatea antrenării și consistența între eșantioane.

Având în vedere că numărul de imagini disponibile pentru fiecare persoană este limitat, am aplicat o serie de transformări aleatorii doar asupra setului de antrenare, cu scopul de a simula variații realiste și de a reduce riscul de supraînvățare. Aceste transformări includ:

- Răsturnare orizontală aleatorie – simulează schimbări de poziționare a feței în cadru;
- Variații subtile de culoare – ajustări minore ale luminozității, contrastului, saturației și nuanței
- Zgomot JPEG controlat – ușoare degradări ale calității imaginii
- Perturbări minore ale detaliilor – prin aplicarea aleatorie a acestor transformări, imaginile devin ușor diferite între epoci, fără a altera conținutul relevant pentru recunoaștere.

Scopul acestor augmentări a fost de a expune modelul la condiții cât mai variate, astfel încât să învețe să recunoască aceleași persoane chiar și în prezența unor diferențe subtile între imagini. Transformările au fost aplicate exclusiv în timpul antrenării, păstrând datele de validare nemodificate, pentru a reflecta performanța reală a modelului.

Prin acest proces, setul de date l-am extins artificial, fără a adăuga imagini noi și îmbunătățind capacitatea de generalizare a modelului la variații nesemnificative.

Generarea perechilor de imagini

Pentru antrenarea unei rețele siameze este necesară utilizarea de perechilor de imagini. Fiecare pereche fiind însoțită de o etichetă care indică dacă cele două imagini reprezintă aceeași persoană sau persoane diferite. Modelul învață să diferențieze între fețele care aparțin aceluiași individ și cele care aparțin unor persoane distincte.

Fiecare pereche conține două imagini care sunt introduse separat prin cele două ramuri convoluționale ale rețelei. În funcție de identitatea persoanelor din pereche, acestea au fost împărțite astfel:

- Perechi pozitive – formate din două imagini diferite ale aceleiași persoane. Acestea ajută modelul să învețe similaritatea dintre trăsături faciale ale aceluiași individ, chiar dacă imaginile diferă ca expresie, lumină sau unghi.

- Perechi negative – sunt formate din două imagini ale unor persoane diferite. Acestea au fost generate în două moduri:
  - o imagine din setul pozitiv și una aleatorie din setul de imagini negative
  - două imagini ale unor persoane diferite din cadrul setului pozitiv

  Fiecare pereche a fost etichetată în felul următor:
- 1 pentru perechile pozitive (aceeași persoană)
- 0 pentru cele negative (persoane diferite)

Pentru a evita influențarea modelului de un tipar nedorit în ordinea datelor, toate perechile generate au fost amestecate aleator. Dacă modelul învăța dintr-o secvență repetitivă de exemple întâi doar perechi pozitive, apoi doar negative, ar putea dezvolta un comportament nedorit sau incorect. Acest comportament este cunoscut sub numele de bias (tendință).

Etichetele asociate fiecărei perechi, numite și labeluri, indică dacă cele două imagini reprezintă aceeași persoană (label 1) sau persoane diferite (label 0). Prin amestecarea perechilor, m-am asigurat că modelul vede o combinație variată de exemple. Am evitat ca modelul să dezvolte o ordine care i-ar putea influența greșit procesul de învățare.

Împărțirea și echilibrarea datelor

Pentru ca modelul să învețe să recunoască persoane noi și să generalizeze bine la date nevăzute, am împărțit setul pozitiv în două subseturi, pe baza identității persoanelor:

- 60% dintre persoane au fost folosite pentru antrenare;
- 40% dintre persoane au fost alocate pentru validare.

Am făcut această alegere din dorința de a păstra un echilibru între cantitatea de date disponibile pentru antrenarea rețelei.

Această împărțire pe identitate și nu pe imagini a fost pentru a mă asigura că modelul nu memorează trăsături ale persoanelor din setul de antrenare. Modelul trebuie să învață să compare fețe în mod general. Astfel, în setul de validare am inclus doar persoane noi, pe care modelul nu le-a „văzut" niciodată în timpul învățării.

După generarea tuturor perechilor (pozitive și negative), am echilibrat datele astfel încât să conțină un număr egal de exemple din ambele clase. Fără această balansare, modelul ar fi putut învăța să favorizeze clasa cu cele mai multe exemple (de regulă, negative). Menținând un raport echilibrat între clase am asigurat o învățare corectă și o evaluare obiectivă a performanței.

Pregătirea finală a datelor

După generarea și etichetarea perechilor, am transformat datele într-un format compatibil cu TensorFlow.

Fiecare lot de antrenare a fost compus din două imagini: una de referință și una de comparație; și eticheta asociată perechii. În cazul setului de antrenare, imaginile le-am augmentat aleator pentru a introduce diversitate. Pe de altă parte, setul de validare l-am păstrat nemodificat pentru a menține realitatea.

Datele au fost grupate în batch-uri de câte 8 exemple. Această valoare am ales-o încât să echilibreze consumul de memorie cu antrenărea. De asemenea, am aplicat și amestecarea datelor (eng. shuffle) și preîncărcarea în memorie (eng. prefetch), care ajută la fluidizarea procesului de antrenare.

În ansamblu, toate aceste etape au contribuit la formarea unui set de date organizat pentru realizarea modelului.

Capitolul 5. Proiectare de detaliu și implementare

5.1. Structura generală a sistemului

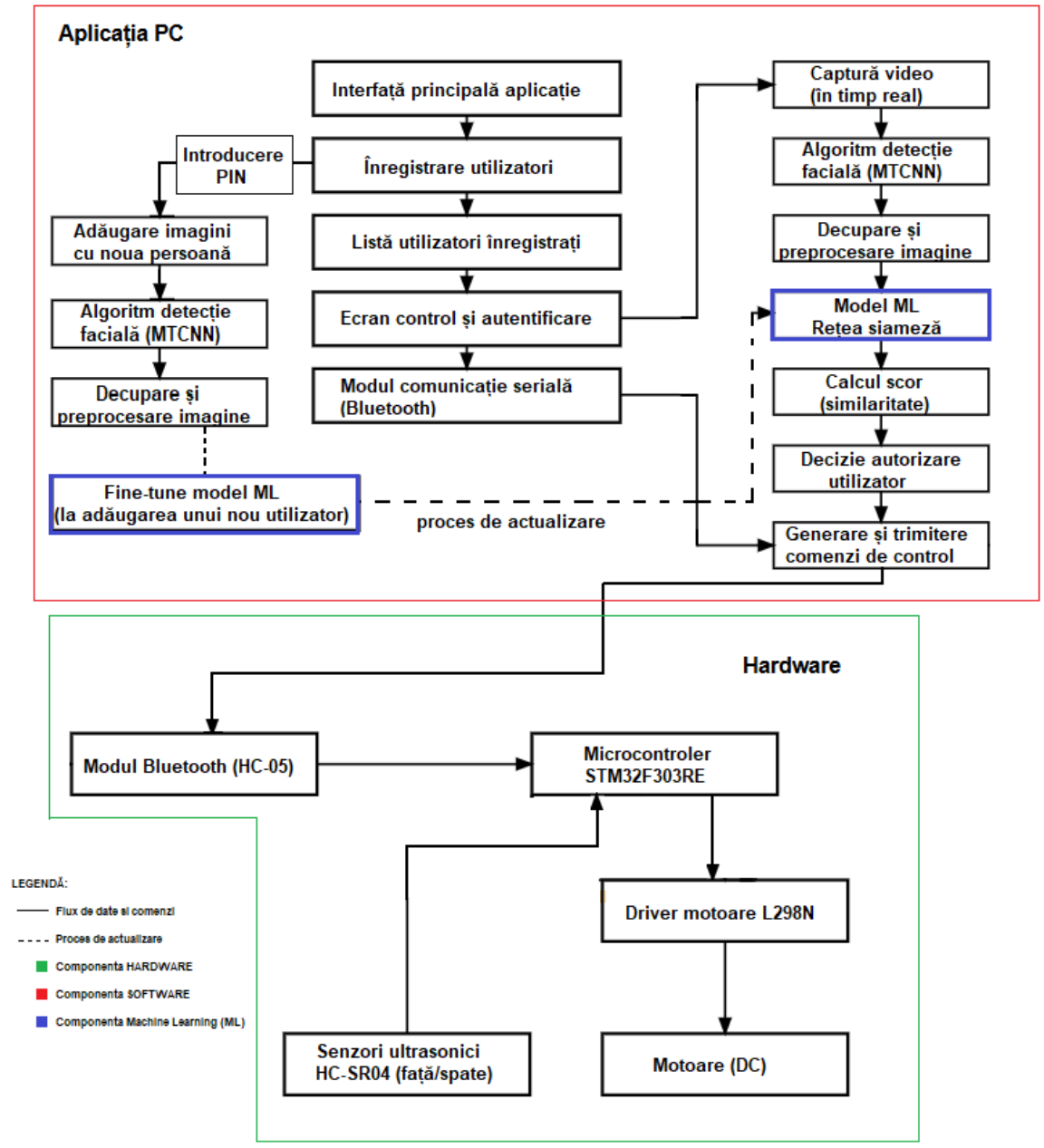


Figura 5.1: Diagrama funcțională a sistemului: interacțiunea dintre aplicație și componenta hardware

În schema generală a sistemului din figura 5.1, pentru o viziune de ansamblu a sistemului am reprezentat modul de funcționare al celor mai importante componente. Din logica sistemului fac parte componenta hardware, componenta software și componenta Machine Learning. Cea din urmă componenta este inglobată în componenta software ce reprezintă aplicația generală. Am expus conexiunile dintre ele, dar și interacțiunile. În contextul diagramei, am surprins cele mai importante module ale aplicației și fluxul de procesare. Am menționat și etapele specifice recunoașterii faciale dezvoltate în modelul meu ce aparține componentei de Machine Learning. O altă mențiune este partea de control a mașinii fizice constituită din microcontroler și module specifice. În următoarea parte a lucrării prezente am detaliat elementele și rolul fiecăruia în cadrul sistemului integrat.

5.2. Componenta hardware

Robotul mașină reprezintă componenta fizică și executivă a sistemului de verificare facială. Ulterior, după ce se autorizează persoana prin identitatea acesteia, la nivelul aplicației, robotul este deschis pentru acceptarea comenzilor de mișcare.

Sistemul hardware al robotului este compus dintr-un set de componente electrice interconectate. Elementele fizice au un rol important în realizarea aplicatiei. Componentele și rolul acestora urmează să fie descrise individual în urmatoarele paragrafe.

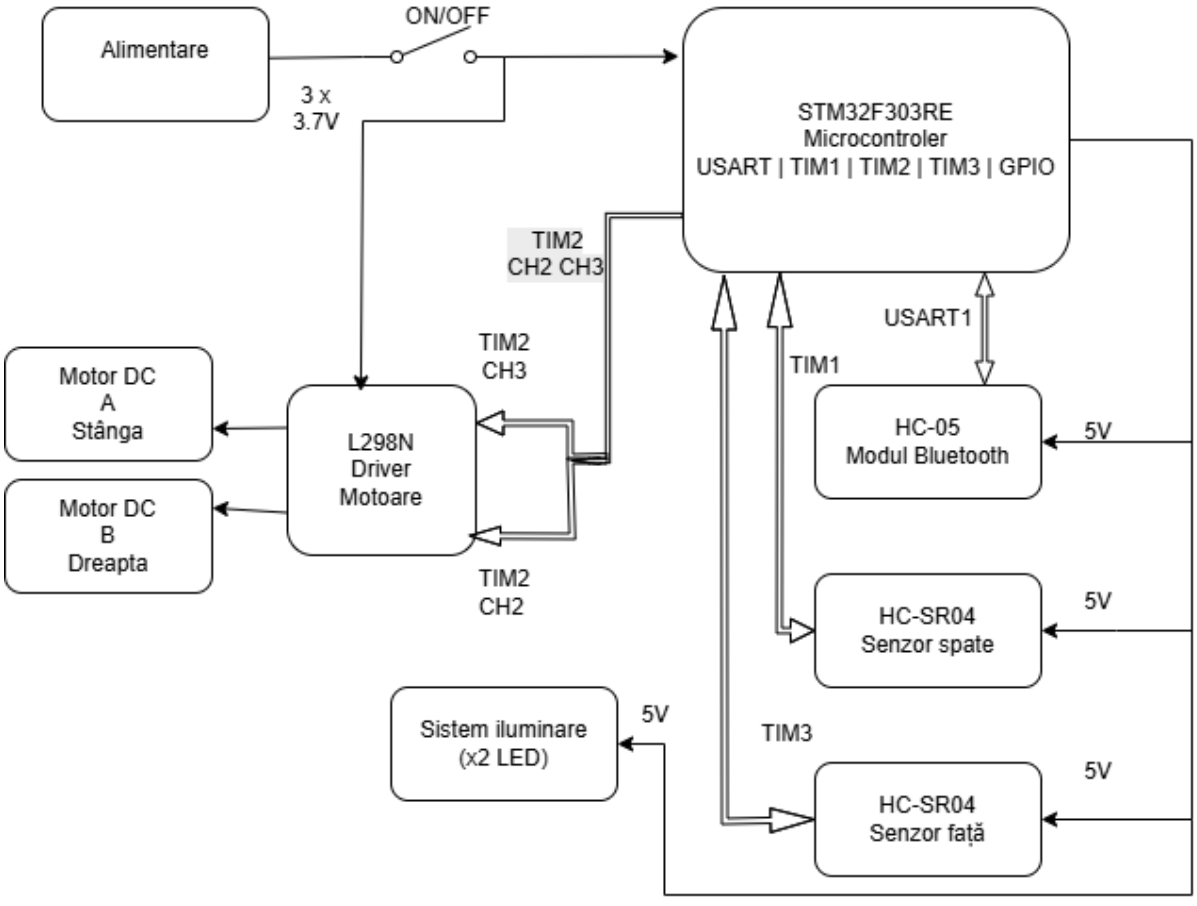


Figura 5.2: Structură logică hardware a microcontrolerului și modulelor periferice [1]

[1]Imagine realizată cu ajutorul platformei online `https://app.diagrams.net/`.

Integrarea componentelor hardware și logica de control firmware

- **Rolul central al microcontrolerului**
  Microcontrolerul STM32F303RE reprezintă centrul de comandă al întregului sistem hardware. Rolurile principale ale microcontrolerului sunt de a interpreta comenzile venite prin conexiunea Bluetooth și gestionarea datelor pentru controlul motoarelor. Primeste informații din mediul exterior de la două module HC-SR04 și le proceseaza pentru a evita obstacolele aflate în calea robotului. Controlează două motoare prin semnalele PWM generate. Acesta rulează firmware-ul dezvoltat în limbajul C cu ajutorul bibliotecii HAL și a mediului de dezvoltare STM32CubeMX. Sunt utilizate mai multe periferice pentru realizarea funcționalităților în nivelul hardware. Modulele GPIO, USART1, TIM1, TIM2 și TIM3 au fost activate în procesul de realizare al sistemului.
- **Asocierea perifericelor cu modulele funcționale**
  Am utilizat mai multe periferice pentru realizarea funcționalităților la nivelul hardware. Modulele le-am activat în procesul de realizare al sistemului. În vederea implementării funcționalităților hardware ale aplicației, microcontrolerul STM32F303RE utilizează următoarele periferice, fiecare asociat direct cu un modul al sistemului:
  - USART1: utilizat pentru comunicarea serială cu modulul Bluetooth HC-05.
  - TIM1: asociat senzorului ultrasonic HC-SR04 din spate.
  - TIM3: asociat senzorului ultrasonic HC-SR04 din față.
  - TIM2: responsabil de generarea semnalelor PWM pentru controlul vitezei celor două motoare DC prin driverul L298N.
  - GPIO: configurat pentru multiple funcții.
- **Integrarea perifericelor interne cu modulele funcționale ale sistemului**

  În cadrul proiectului, pinii **GPIO** (General Purpose Input/Output) ai microcontrolerului STM32 sunt configurați pentru controlul direcției motoarelor, trimiterea datelor către acestea și primirea de impulsuri de la senzorul ultrasonic HC-SR04. De asemenea, o parte dintre acești pini sunt folosiți pentru comunicarea serială cu modulul Bluetooth HC-05, prin conectarea la interfața USART1, unde pinii PA9 (TX) și PA10 (RX) sunt configurați ca ieșiri și respectiv intrări digitale pentru transmisia și recepția datelor.

  Comunicarea se face prin intermediul protocolului de comunicație serială USART pentru transmiterea datelor intre două dispozitive. Am folosit USART1 ca interfață serială integrată în microcontroler pentru transmiterea de date bit cu bit. Comunicarea este asincronă fără ceas comun, în cazul acesta de utilizare. USART1 este parte internă și una dintre cele mai rapide interfețe conectate la magistrala de periferice ale unității. Această interfața permite comunicarea cu dispozitive independente precum senzorii și modulul Bluetooth folosind doar două fire TX(transmitere) și RX(recepție).

  USART1, parte înglobată în microcontroler, este utilizata pentru comunicația serială cu modulul Bluetooth. Modulul HC-05 transmite comenzile de control preluate de la aplicația principală. Comenzile sunt interceptate de modul și transmise mai departe la microcontroler. USART1 este configurat la o rată de transfer (baud rate) de 9600, cu 8 biți de date, fără paritate, un bit de stop și în modul asincron full duplex. Rata de transfer sau viteza de transmisie indică numarul de biți transmiși pe secundă. Biții sunt tranșmisi printr-un canal de comunicație seriala. Configu-

rarea cu 8 biți de date constă în transmiterea unui caracter prin interfața seriala reprezentat pe 8 biți respectiv un octet. Standardul folosit permite transmiterea valorii ASCII (American Standard Code for Information Interchange) a caracterelor primite de la aplicația centrală către microcontroler.

Inițializarea interfației USART1 se realizează cu biblioteca HAL, prin functia utilizată "MX_USART1_UART_Init()" și setează parametrii de configurare menționați mai sus.

Am optat pentru utilizarea modului interrupt-based pentru receptia non-blocantă a caracterelor. Atunci când un caracter este recepționat, se declanșează automat o întrerupere. Funcția "HAL_UART_RxCpltCallback()" este apelată simultan cu recepționarea întreruperii. Funcția verifică și procesează datele primite, ulterior este reapelată funcția "HAL_UART_Receive_IT()" pentru o comunicare continuă. Termenul "full duplex" sugerează transmiterea și recepția simultană a datelor prin două linii separate transmisie (TX), respectiv recepție (RX). Comunicația asincrona nu presupune un semnal de ceas între cele două dispozitive. Sincronizarea se face prin viteza de transmisie stabilită, bitul de start și de stop, pentru delimitarea fiecărui caracter transmis.
Montajul fizic l-am realizat în corespondență cu tabelul (5.1).

Tabela 5.1: Maparea fizică a pinilor pentru HC-05

Funcție	Denumire pin STM32	Conector placă (CN)	HC-05
USART1_TX	PC4	CN10 - pin 34	pinul RX
USART1_RX	PC5	CN10 - pin 6	pinul TX
5V	-	CN6 - pin +5V	pinul VCC
GND	-	CN6 - GND	pinul GND

Prin interfața descrisă, comenzile sunt trimise în timp real la microcontroler, prin tehnologia fără fir a modulului HC-05.

Părțile responsabile cu detectarea obiectelor din proximitatea robotului sunt doi senzori ultrasonici de tip HC-SR04. Am utilizat în dezvoltare doi astfel de senzori: unul montat în partea frontală a robotului, iar celălalt în partea posterioară a acestuia.

Fiecare senzor este controlat de un timer intern al microcontrolerului. Am ales **TIM1** și **TIM3** pentru realizarea funcționalității propuse. Timerele sunt configurate în modul de bază (Base Timer în STM32CubeMX) pentru a genera impulsurile necesare pentru măsurarea distanței până la un obiect.

Am configurat cu un prescaler de 71 și o perioadă de 1999.

Un prag de 10 cm am setat pentru ambii senzori, și reprezintă distanța minima menținută intre un obiect și robotul. În momentul depașirii limitei stabilite senzorul detectează și trimite spre microcontroler impulsuri. Timerele asociate zonelor din față (TIM3) și în spate (TIM1) inițializează un impuls scurt de 10µs pe pinul TRIG către senzorul ultrasonic. Senzorul emite la rândul lui un semnal ultrasonic în exteriorul robotului. Pinul ECHO așteaptă pâna la recepționarea semnalului. Când semnalul ajunge la terminalul de recepție al senzorului, se începe măsurarea timpului de întoarcere. Dacă pragul de 10 cm a fost depășit, se va trimite informația spre microcontroler. Montajul fizic s-a realizat conform cu tabelul 5.2 pentru HC-SR04 față, respectiv 5.3, pentru partea din spate.

Motoarele pe care le-am utilizat în proiect sunt două motoare de curent continuu (DC). Acestea asigură deplasarea robotului înainte și înapoi. Sunt controlate cu ajutorul

Tabela 5.2: Maparea fizică a pinilor pentru senzorul HC-SR04 (față)

Funcție	Denumire pin STM32	Conector placă (CN)	HC-SR04 Față
GPIO_Output	PA9	CN5 - pin 1	pinul TRIG
GPIO_Input	PA8	CN9 - pin 8	pinul ECHO
5V	—	CN6 - +5V	pinul VCC
GND	—	CN6 - GND	pinul GND

Tabela 5.3: Maparea fizică a pinilor pentru senzorul HC-SR04 (spate)

Funcție	Denumire pin STM32	Conector placă (CN)	HC-SR04 Spate
GPIO_Output	PA5	CN5 - pin 6	pinul TRIG
GPIO_Input	PA6	CN5 - pin 5	pinul ECHO
5V	—	CN6 - +5V	pinul VCC
GND	—	CN6 - GND	pinul GND

unui driver de motoare compatibil, conectat între microcontrolerul STM32F303RE și cele două motoare. O funcție a driverului este să preia semnalele de control generate de placă și să asigure curentul necesar pentru mișcarea motoarelor.

Reglarea vitezei motoarelor, se face prin semnale PWM (Pulse Width Modulation) generate de perifericul **TIM2** al microcontrolerului STM32. Timerul l-am configurat în STM32CubeMX pentru a putea lucra la o frecvență de aproximativ 20 kHz. Din TIM2 au fost utilizate două ieșiri: PA1 pentru canalul 2 (CH2), care controlează motorul B (dreapta) prin pinul ENB, și PB10 pentru canalul 3 (CH3), care controlează motorul A (stânga) prin pinul ENA. Acești pini au fost configurați pentru a emite semnale PWM către intrările de control ale driverului. Controlul vitezei și direcției motoarelor se face prin ajustarea coeficientului „duty cycle" semnalului PWM în funcție de comenzile primite prin modulul Bluetooth HC-05 și în funcție de datele primite de la senzorii de distanță (HC-SR04). Detectarea unui obstacol în apropiere, sub pragul definit (10 cm) rezultă modificarea semnalului de PWM respectiv oprirea automată a roților.

Driverul este alimentat de la sursa principală a sistemului, iar intrările ENA (PB10) și ENB (PA1) controlează activarea fiecărui motor. Direcția fiecărui motor este determinată prin starea logică a pinilor IN1–IN4, astfel: pentru motorul A (stânga), s-au folosit PB5 (IN1) și PB4 (IN2), iar pentru motorul B (dreapta), PB0 (IN3) și PA4 (IN4). Conexiunile fizice spre motoare sunt următoarele: pentru motorul B, OUT4 (fir roșu) este conectat la borna de jos, iar OUT3 (fir negru) la borna de sus; pentru motorul A, OUT1 (fir roșu) este conectat la borna de jos, iar OUT2 (fir negru) la borna de sus a motorului. Vizualizare detaliată în tabelul (5.4).

Pentru o mai bună înțelegere a conexiunilor fizice dintre driverul L298N și motoarele robotului am realizat un tabel descriptiv. Am pus în lumină modul prin care fiecare pin de ieșire al driverului este conectat la bornele motoarelor. În funcție de direcție și polaritate sunt reprezentate prin culorile firelor de conexiune. Conexiunile sunt evidențiate în montajul fizic după cum am menționat în tabelul dat (5.6).

Sensul de rotație este determinat de combinația stărilor logice HIGH sau LOW. Aplicate pe pinii de intrare ai fiecărui motor. De exemplu, pentru a avansa, motorul A primește semnal logic 1 pe IN1 și 0 pe IN2, iar motorul B primește 0 pe IN3 și 1 pe IN4. Inversarea semnalelor determină mersul înapoi. În tabelul (5.5) este ilustrată corespondența dintre stările pinilor și direcția de rotație pentru fiecare motor.

Tabela 5.4: Maparea pinilor STM32 pentru controlul motoarelor prin driverul L298N

Funcție	Pin STM32	Conector placă (CN)	Pin driver L298N
Activare motor A	PB10	CN9 - pin 7	ENA
Direcție motor A	PB5	CN9 - pin 5	IN1
Direcție motor A	PB4	CN9 - pin 6	IN2
Direcție motor B	PB0	CN8 - pin 4	IN3
Direcție motor B	PA4	CN8 - pin 3	IN4
Activare motor B	PA1	CN8 - pin 2	ENB
Alimentare driver	—	-	+12V
Masa comună	—	CN6 - GND	GND

Tabela 5.5: Stările pinilor de control și direcția de rotație a motoarelor

Motor	Stare pin 1	Stare pin 2	Sens de rotație
Motor A (IN1 = PB5, IN2 = PB4)	1 (IN1)	0 (IN2)	Înainte
	0 (IN1)	1 (IN2)	Înapoi
	0 (IN1)	0 (IN2)	Stop
Motor B (IN3 = PB0, IN4 = PA4)	0 (IN3)	1 (IN4)	Înainte
	1 (IN3)	0 (IN4)	Înapoi
	0 (IN3)	0 (IN4)	Stop

Pentru controlul vitezei de rotație a motoarelor, în proiect am utilizat semnalul PWM generat de timerul TIM2 al microcontrolerului. În implementare, valoarea ciclului a fost setată la aproximativ 25% din perioada totală, corespunzător unei valori de 500 din 2000 (prescaler 71 și o perioadă de 1999), ceea ce înseamnă o viteză moderată a motoarelor. Viteza definită impune un control fin al deplasării, evită accelerațiile bruște și facilitează oprirea rapidă. 5.7

În figura (5.3) am prezentat distribuția pinilor pentru placa NUCLEO-F303RE, după configurarea perifericelor în aplicația STM32CubeMX. Această reprezentare a fost generată automat în în urma selecției interfețelor utilizate în proiect (UART, PWM, GPIO etc.). Este o vedere de ansamblu asupra pinilor activi și a funcțiilor asociate acestora. Distribuția completă a tuturor pinilor este prezentată în Anexa A – figura A.1, conform documentației oficiale a plăcii.

Alimentarea componentelor hardware ale sistemului

- Robotul este alimentat de trei acumulatori de 3.7V conectați în serie. Acești acumulatori sunt sursa principală de alimentare a sistemului. Împreună, aceștia oferă o tensiune totală de aproximativ 12V. Această tensiune este suficientă pentru a porni atât motoarele, cât și placa de dezvoltare.
- Placa de dezvoltare NUCLEO-F303RE am configurat-o să fie alimentată extern din sursa principală a sistemului, prin pinii VIN sau E5V, conform poziționării jumperului JP5. Jumperul JP5 este poziționat pentru a permite alimentarea plăcii direct din sursa externă conectată la pinul VIN. Această configurare a fost realizată pentru a permite plăcii să funcționeze independent față de portul USB al calculatorului. În figura A.2 din Anexa A este prezentată o reprezentare grafică a plăcii NUCLEO-F303RE. În schema oferită sunt evidențiați pinii și modul de configurare a jumperului JP5.

Tabela 5.6: Conexiuni fizice dintre driverul L298N și motoare

Pin driver L298N	Culoare fir	Conexiune motor
OUT1	Roșu	Motor A – borna de jos
OUT2	Negru	Motor A – borna de sus
OUT3	Negru	Motor B – borna de sus
OUT4	Roșu	Motor B – borna de jos

Tabela 5.7: Corelația dintre valoarea semnalului PWM și viteza motoarelor

Valoare PWM (duty cycle)	Viteză motor
0 (0%)	0% (motor oprit)
500 (25%)	Viteză redusă
1000 (50%)	Viteză medie
1500 (75%)	Viteză mare
2000 (100%)	Viteză maximă

- Pe driverul L298N, tensiunea este distribuită direct de la sursa principală a sistemului către cele două motoare. Driverul preia alimentarea și o împarte prin cele două canale independente ale sale Fiecare canal furnizează tensiunea necesară unui motor DC pentru funcționare.
- Modulul Bluetooth HC-05 este alimentat de la ieșirea de 5V a plăcii NUCLEO. Tensiunea oferită este suficientă pentru ca modulul să funcționeze stabil și să comunice cu microcontrolerul prin conexiunea serială.
- Senzorii ultrasonici HC-SR04 (cel din față și cel din spate) sunt conectați tot la sursa de 5V a plăcii. Aceștia au nevoie de această tensiune pentru a putea trimite și primi corect semnalul ultrasonic folosit pentru măsurarea distanței.
- În circuit sunt incluse două LED-uri albastre, conectate la sursa de 5V a microcontrolerului prin rezistențe de 220 Ω, fiecare. Acestea sunt montate în partea frontală a robotului și sunt folosite ca faruri.

Toate componentele sistemului sunt conectate la o masă comună (GND).

Configurarea modulului Bluetooth HC-05

Modulul Bluetooth HC-05 a fost configurat cu ajutorul aplicației Tera Term, folosind un adaptor USB-TTL. Am realizat următoarele operații prin intermediul aplicației Tera Term:

1. Deschiderea aplicației Tera Term și selectarea portului serial (COM10).
2. Configurarea parametrilor de comunicare: baud rate 38400, 8 biți de date, fără paritate, 1 bit de stop, fără control de flux.
3. Accesarea modului AT și trimiterea comenzilor:
   - AT – am verificat conexiunea și am primit raspunusl ”OK”
   - AT+ROLE=0 rolul l-am definit slave,
   - AT+NAME=HC-05-licenta:am redenumit modulul
   - AT+PSWD=2025: am setat parola

În figura 5.5 am ilustrat pașii efectuați în interfața Tera Term pentru configurarea modulului Bluetooth HC-05 în modul AT.

[2]Imagine generata din STM32CubeMX dupa configurarea perifericelor.

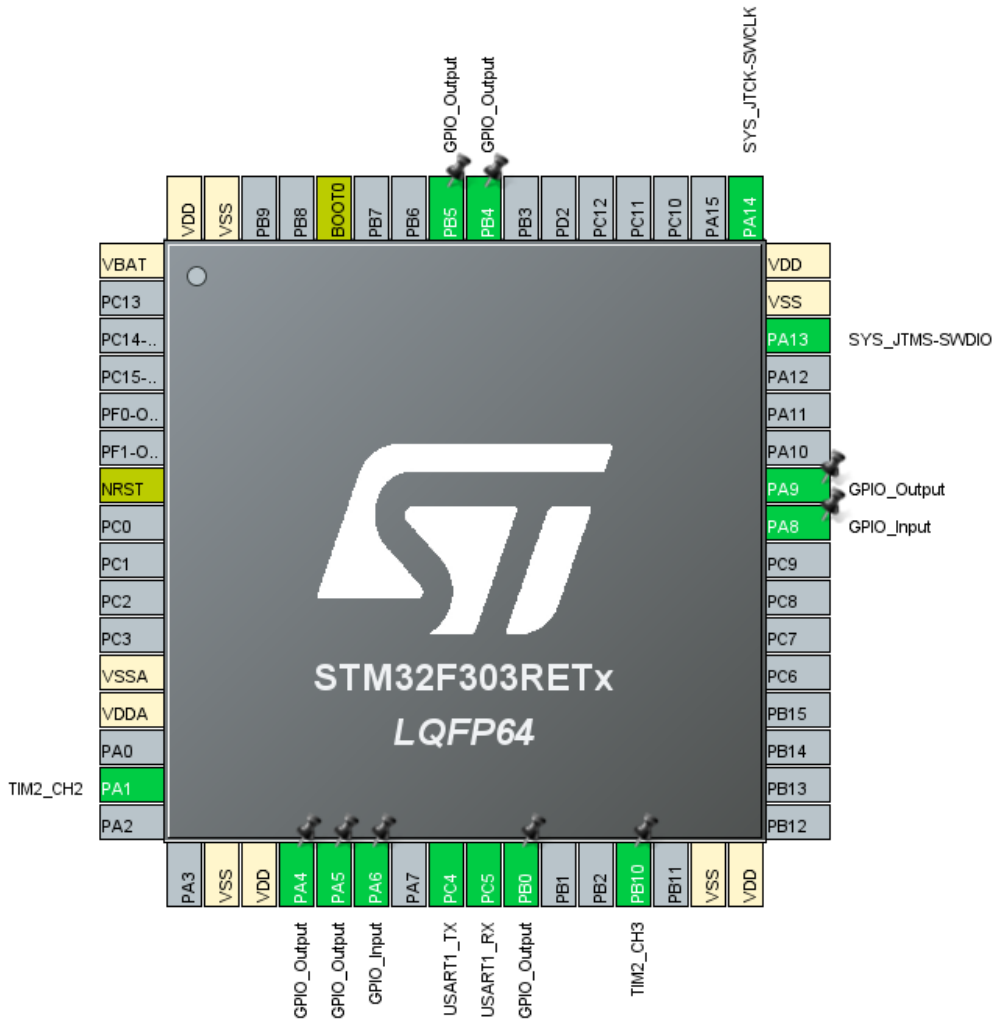


Figura 5.3: Distribuția pinilor pe placa NUCLEO-F303RE. [2]

Conexiunile necesare pentru cele două dispozitive se pot vedea în figura (5.4).

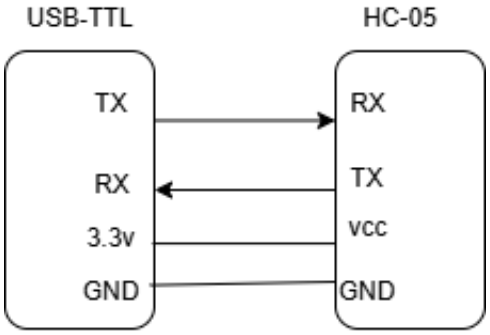


Figura 5.4: Conexiuni HC-05 și adaptorul USB-TTL [3]

[3]Imagine realizată cu ajutorul platformei online `https://app.diagrams.net/`.

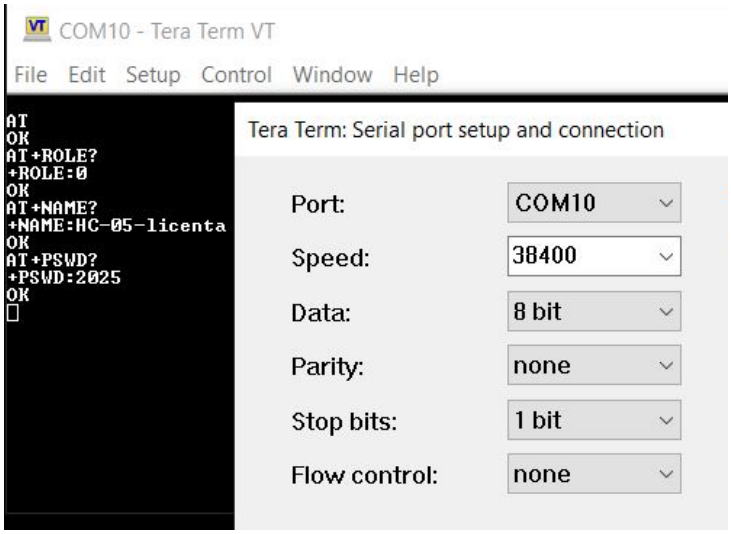


Figura 5.5: Configurarea modulului HC-05 în Tera Term

5.3. Componenta Machine Learning

Dinamică generală

Pentru a explica cât mai bine cum am implementat verificarea facială în aplicație, am reprezentat în figura 5.6 pașii logici parcurși. Procesul începe cu imaginile brute și se încheie cu decizia finală oferită de algoritmul de Machine Learning. Fiecare etapă am documentat-o în urmatoarele subsecțiuni ale lucrarii.

Modelul primește ca intrare două imagini: una este considerată imagine de referință (ancoră), iar cealaltă este imaginea test. În prima etapă, ambele trec printr-un proces de detecție facială automatizată folosind algoritmul MTCNN, urmat de decupare și redimensionare la o dimensiune fixă de 105×105 pixeli.

În etapa următoare am normalizat valorile pixelilor în intervalul $[0, 1]$. Pentru imaginile din setul de antrenare am aplicat și augmentări aleatorii. Am aplicat inversarea orizontală și variații subtile de culoare.

Ulterior, sunt formate perechi de imagini, pozitive: imagini ale aceleiași persoane și negative: imagini ale unor persoane diferite. Perechilor le-am atașat eticheta 0 sau 1 în funcție de identitate. Aceste perechi le-am utilizat ca date de intrare pentru rețeaua siameză.

Rețeaua are două ramuri CNN identice, care extrag vectori de trăsături (embedding-uri) pentru fiecare imagine. Apoi, diferența absolută dintre acești vectori este calculată și transmisă unui strat dens cu activare sigmoid. Rezultatul este un scor de similaritate între 0 și 1, și indică cât de apropiate sunt fețele comparate.

Pe baza acestui scor, aplicația decide dacă imaginile aparțin aceleiași persoane. Dacă scorul este mai mare decât un prag stabilit, în cazul meu 0.5, sistemul le consideră similare și returnează 1. În caz contrar, se returnează 0, indicând că persoanele sunt diferite.

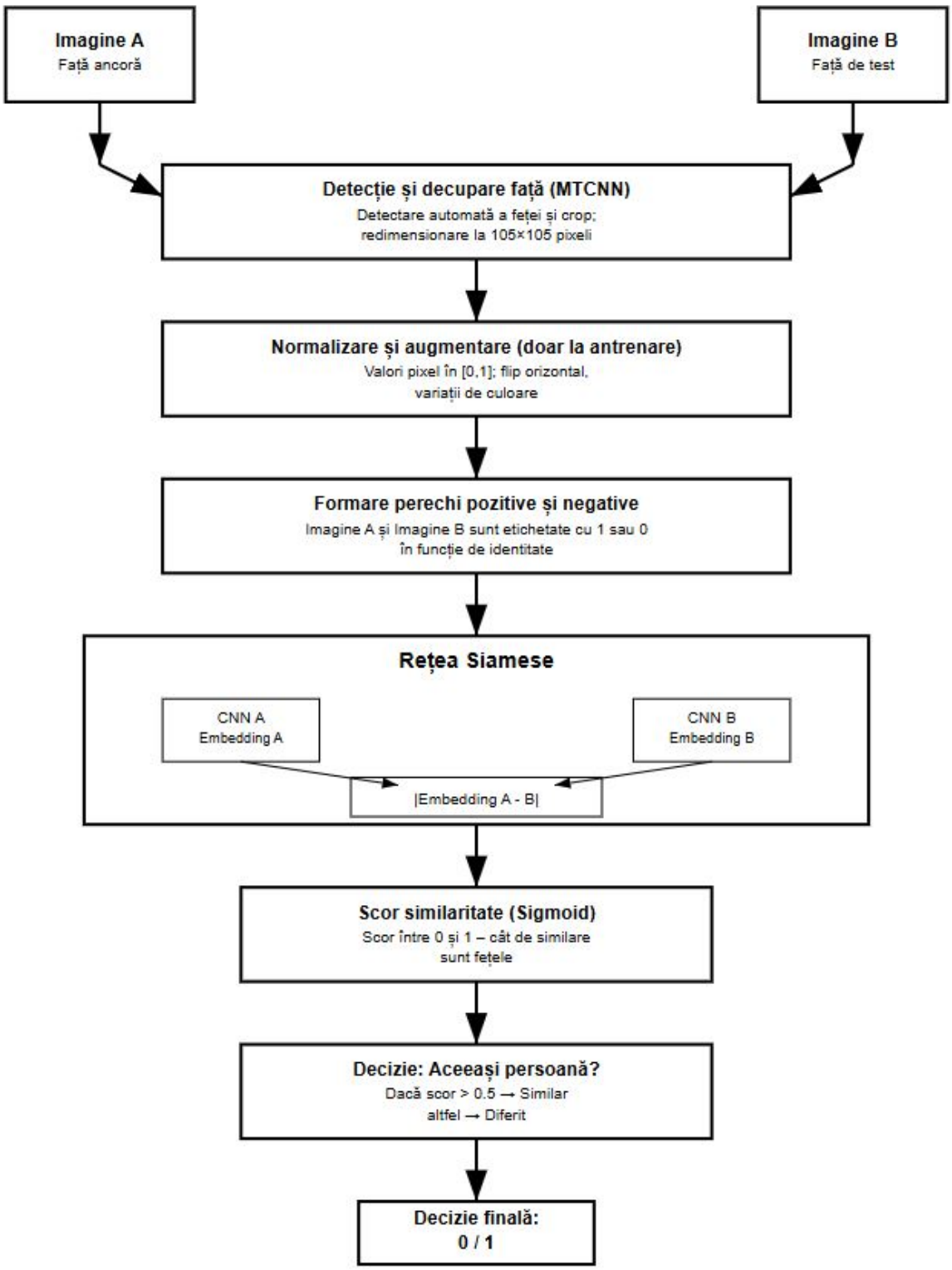


Figura 5.6: Fluxul general al modeului pentru verificare facială

Rețeaua siameză

Rețeaua siameză reprezintă componenta centrală a modelului. Este responsabilă de compararea trăsăturilor extrase din cele două imagini introduse în sistem. Particularitatea acestui tip de rețea constă în faptul că utilizează două ramuri identice, cu aceleași

ponderi, pentru a procesa imaginile de intrare în paralel.

Structura generală este prezentată în figura 5.7. Fiecare imagine, după ce a fost decupată și preprocesată, este transmisă către un model convoluțional identic, care extrage din ea un vector numeric – numit vector de trăsături. Acest vector este o reprezentare compactă a feței și are rolul de a surprinde caracteristicile distinctive ale acesteia.

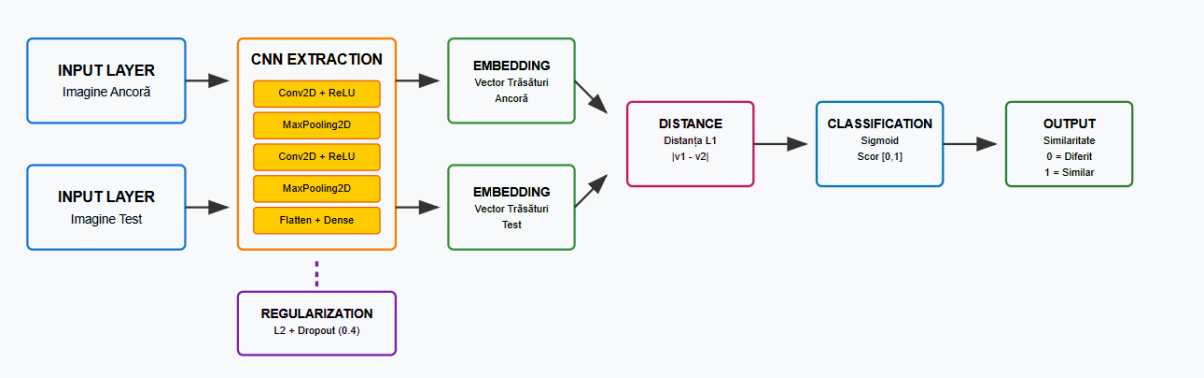


Figura 5.7: Structura rețelei siameze utilizate

După extragerea embedding-urilor pentru cele două imagini, acestea sunt comparate prin calculul diferenței absolute între componentele lor. Această diferență este apoi transmisă către un strat dens, care aplică o funcție de activare sigmoid și returnează un scor între 0 și 1. Scorul obținut reflectă gradul de similaritate între cele două fețe analizate.

Un scor apropiat de 1 indică faptul că cele două imagini sunt cel mai probabil ale aceleiași persoane, în timp ce valori apropiate de 0 sugerează că persoanele sunt diferite. Pragul de decizie poate fi ajustat în funcție de sensibilitatea dorită.

Prin această abordare, am încercat ca rețeaua să învețe să recunoască similarități între persoane fără a avea nevoie de o clasificare explicită a identităților. În esență, a devenit capabil să decidă dacă două fețe aparțin aceleiași persoane, indiferent de poziție, expresie sau iluminare.

Arhitectura rețelei CNN

Fiecare ramură a rețelei siameze este construită pe o rețea convoluțională identică (CNN), responsabilă cu extragerea caracteristicilor esențiale din imagini. Structura acestei rețele este ilustrată în figura 5.8.

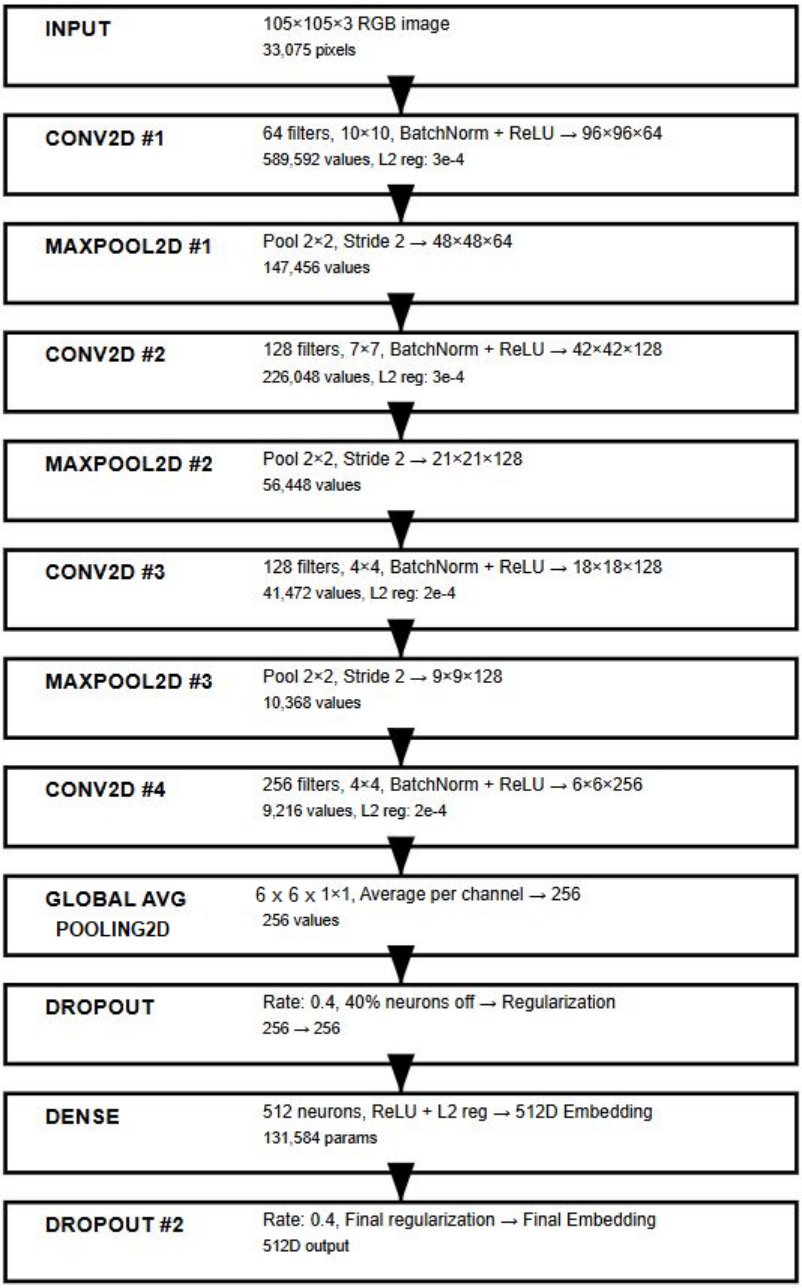


Figura 5.8: Structura CNN utilizată pentru extragerea vectorului de trăsături

Am trimis spre intrare o imagine de dimensiunea 105×105 și trei canale RGB (Red Green Blue). Am ales această dimensiune pentru a echilibra complexitatea modelului cu performanța de procesare. Ca primă operație am aplicat o convoluție cu 64 de filtre de dimensiune 10×10, urmată de normalizare prin BatchNorm și o activare ReLU. După fiecare strat convoluțional, dimensiunile spațiale le-am redus prin straturi de tip MaxPooling2D.

Ulterior, numărul de filtre l-am crescut treptat la 128 și apoi la 256. Dimensiunea kernelului scade și permite captarea unor detaliilor. Fiecare strat convoluțional l-am regularizat printr-un procent de penalizare L2, astfel reduc riscul de suprainvățare.

Am redus total dimensiunile spațiale după ultimele convoluții, cu un strat de GlobalAveragePooling2D, ceea ce conduce la un vector unidimensional. Fiecare valoare co-

respunde unui canal în acest vector.

După această etapă, datele le-am trecut printr-un strat de Dropout cu rată de 0.4, care dezactivează aleator o parte din neuroni la fiecare pas de antrenare. Spre final, informația am transmis-o spre un strat dens format din 512 neuroni. Stratul aplică o activare ReLU. Acest strat produce vectorul final de trăsături.

Diagrama de clase

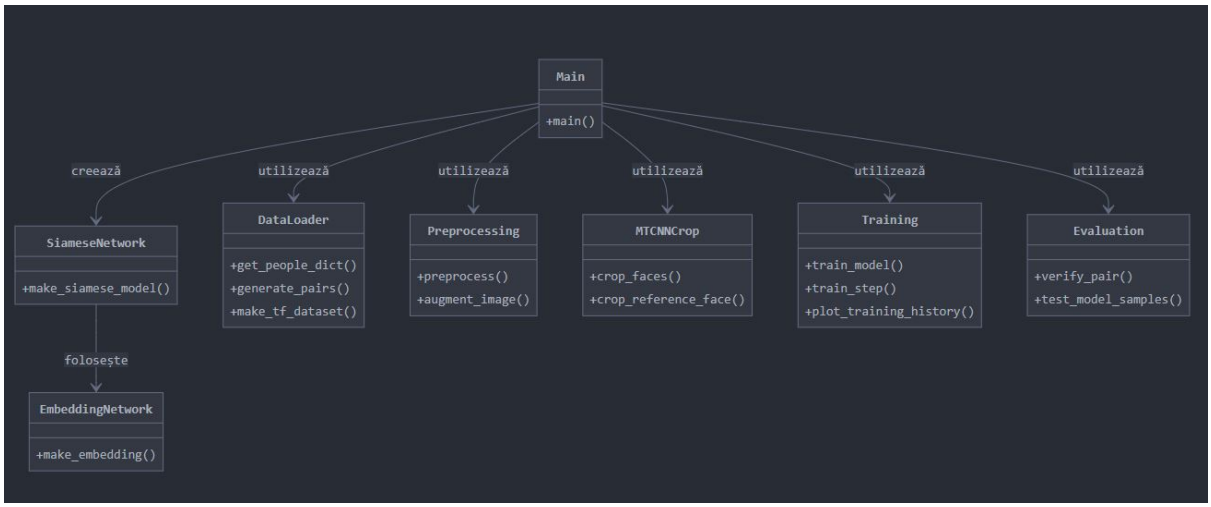


Figura 5.9: Diagrama de clase a aplicației de recunoaștere facială cu rețea siameză.

Am realizat această diagramă de clase pentru a reprezenta principalele componente ale aplicației mele de recunoaștere facială folosind o rețea siameză. În diagrama dată am arătat cum funcția principală, main, coordonează întregul proces, de la încărcarea și preprocesarea datelor, până la antrenarea și evaluarea modelului. Am inclus clasele și funcțiile esențiale, precum și relațiile dintre ele. Am evidențiat modul în care interacționează modulele aplicației de inteligență artificială între ele. 5.9.

5.4. Componenta software

Schema generală a aplicației

Această componentă constă într-o aplicație ce a fost realizată cu scopul de a controla accesul la un sistem fizic pe baza recunoașterii faciale. Am implementat o interfață grafică, prin care utilizatorul poate vizualiza imaginea de la cameră, poate trimite comenzi și poate adăuga persoane noi în sistem. Schema logică a componentei este reprezentată în figura 5.10.

Funcționalitățile aplicației

Am dezvoltat o interfață intuitivă prin care utilizatorul, prin identitatea sa poate controla o mașină inteligentă. La pornire, ecranul principal apare automat, activează camera web a laptopului și afișează comenzile pentru control. Camera începe imediat să capteze imaginea în timp real, iar aplicația analizează constant cadrul pentru a detecta o față. Pentru acest lucru, am folosit algoritmul MTCNN, datorită preciziei în detectarea facială.

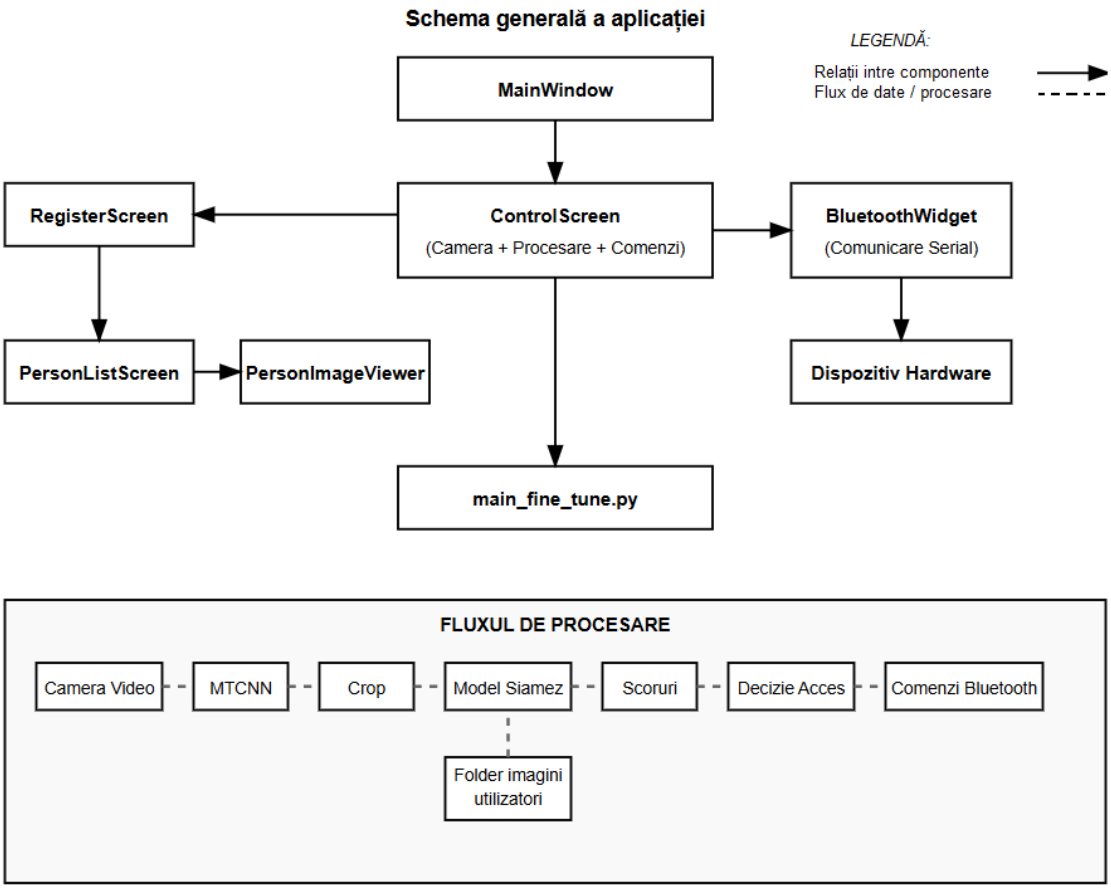


Figura 5.10: Arhitectura logică și fluxul de procesare al aplicației software

Dacă în cadrul camerei este identificată o față, aceasta este decupată și transmisă către modelul de recunoaștere. Verifică se face din partea modelului antrenat, dacă această față este recunoscută. Verificarea persoanei, în cauză, se face prin comparare cu fețele deja existente în sistem. Dacă scorul de asemănare este mare, sistemul consideră că persoana este autorizată. În acest caz, se deblochează comenzile, care pot fi trimise către masină prin Bluetooth prin modul de comunicare serială integrat în aplicație.

Pe de altă parte, dacă fața nu este recunoscută sau nu se găsește nicio situație, accesul este refuzat automat, iar comenzile nu pot fi folosite. În acest punct, utilizatorul are opțiunea de a merge în zona de înregistrare, accesibilă doar cu parolă, unde poate adăuga un nou utilizator. Procesul de înregistrare a unui utilizator este următorul:se introduce un cod PIN și se încarcă una sau mai multe imagini ale persoanei noi. După salvare, aplicația oferă posibilitatea de a actualiza modelul prin procesul de reantrenare.

Această reantrenare, procesul de fine-tuning, este declanșată printr-un script extern care rulează direct din aplicație. Când procesul se termină, utilizatorul primește o notificare. În final, noua persoană poate fi recunoscută de sistem la fel ca cele existente.

Descrierea componentelor implementate

În această secțiune o să fie prezentate modulele și clasele care alcătuiesc aplicația software. Structura modulară ilustrată în figura 5.11 reflectă gestionarea tuturor componentelor: a interfeței, a procesării imaginilor, a recunoașterii faciale și a comunicării cu

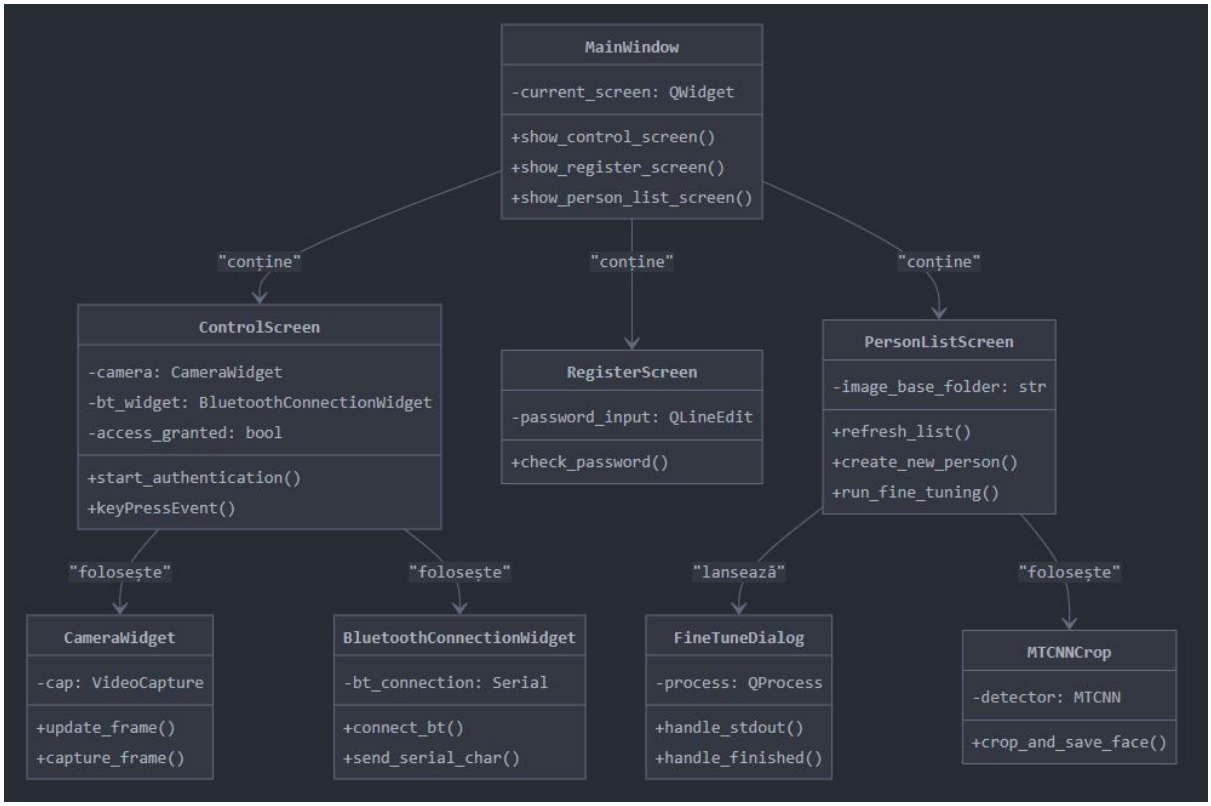


Figura 5.11: Diagrama de clase a aplicației software

dispozitivul fizic. Elementele sunt următoarele:

- MainWindow: Reprezintă clasa principală care pornește interfața grafică, decide ce ecran se afișează: control, înregistrare sau lista persoanelor.
- BluetoothConnectionWidget: Se ocupă cu legătura prin Bluetooth, deschide portul ales și trimite comenzi în format simplu, pentru fiecare acțiune.
- RegisterScreen: Este fereastra unde se adaugă un utilizator nou. Accesul este protejat printr-un PIN. După ce se introduce PIN-ul, se fac capturile și începe procesul de reantrenare al modelului.
- PersonListScreen: Afișează utilizatorii deja existenți. De aici pot fi văzute imaginile asociate fiecăruia sau adăugate unele noi.
- PersonImageViewer Este o componentă care permite vizualizarea imaginilor pentru fiecare persoană. Face parte din fereastra de listă.
- camera_widget: Prelucrează imaginile video de la cameră. Folosește OpenCV pentru a obține cadrele și le trimite spre afișare în aplicație.
- mtcnn_crop: Detectarea feței și decuparea automată au loc în aceast modul prin algoritmul MTCNN.
- result_fine_tune_dialog Aceasta este fereastra care apare în timpul reantrenării modelului. Afișează progresul și din spate automat s-a terminat procesul.

Integrarea modelului de Machine Learning

În aplicația finală, componenta de machine learning este reprezentată de o rețea neuronală siameză. Are rolul de a verifica identitatea utilizatorului pe baza trăsaturilor faciale. Antrenarea modelului se bazează pe Învățatra diferențele și asemănările dintre două imagini. Când o persoană se prezintă în fața camerei, sistemul capturează o ima-

gine în timp real și detectează fața folosind algoritmul MTCNN. Imaginea rezultată din decupaj se va redimensiona și preprocesa, pentru ca trebuie să se respecte formatul de intrare al modelului.

Procesarea constă în trimiterea imaginii catre rețeaua siameză, ce va extrage un vector unic în format numeric. Vectorul este un set de trăsături principale ale feței. Acest vector este comparat cu ceilalți vectori, salvați anterior ai persoanele aparținătoare sistemului. Folosind distanța L1 reprezentată de suma valorilor absolute ale diferențelor între componente, este modalitatea de comparație. Această distanță este trecută printr-o funcție sigmoid și returnează un scor de similaritate între 0 și 1. Dacă acest scor este mai mare decât un pragul pe care l-am ales, utilizatorul este recunoscut și îi e permis accesul.

Performanța sistemului și pentru a-l adapta la fețele utilizatorilor noi, care se vor adaugă, am integrat un proces de fine-tuning. Procesul l-am implementat să pornească automat când se va salvează un nou utilizator în sistem. În contextul reînoirii zonei dedicată imaginilor cu persoane, o persoană autorizată poate sa actualizeze daca și numai dacă introduce un cod PIN valid. PIN-ul pe care l-am ales este „1234”. Aplicația folosește un obiect QProcess din PyQt5 pentru a lansa în fundal scriptul dedicat modelului și îl reantrenează cu noile imagini. Modelul actualizat este spre final salvat și devine funcțional fără intervenția manuală a dezvoltatorului.

Diagrama de secvență

În Figura 5.12 am ilustrat, diagrama de secvență care arată cum funcționează procesul de verificare și control pentru mașina inteligentă. Am realizat această diagramă pentru a evidenția vizual, cum interacționează principalele componente software din aplicație: interfața grafică , camera, algoritmul de detecție facială (MTCNN), modelul de recunoaștere facială, modulul de Bluetooth și, la final, dispozitivul pe care îl controlez. Scenariul începe când utilizatorul apasă pe butonul de START al aplicației. Sistemul verifică dacă Bluetooth-ul este conectat, apoi preia o imagine, caută o față și verifică dacă utilizatorul e înregistrat. Dacă totul este în regulă și scorul obținut este de trecere, aplicația permite controlul mașinii și transmite comenzile mai departe. Dacă nu, accesul este blocat. Am considerat că e utilă vizualizarea aceastei diagrame pentru a înțelege cum funcționeaza aplicația și ce decizii automate se petrec.

Diagrama cazurilor de utilizare ale aplicației

În diagrama de cazurilor de utilizare prezentată în Figura (5.13), am ilustrat funcționalitățile principale pe care le oferă aplicația mea, și relația dintre actori și aceste cazuri de utilizare.

Am identificat doi actori principali: utilizatorul, se va autentifica folosind verificarea facială și administratorul, el are acces la operații de adăugarea sau ștergerea de utilizatori. Numai administratorul poate face reantrenarea modelului de verificare facială.

Pentru fiecare caz de utilizare, am evidențiat și pașii secundari esențiali, cum ar fi controlul mașinii inteligente după autentificare, afișarea unui mesaj de eroare în cazul unui eșec la autentificare sau încărcarea și salvarea datelor pentru un utilizator nou. Am structurat această diagramă pentru a evidenția clar funcțiile principale, respectiv diferențierea între drepturile utilizatorului și cele ale administratorului.

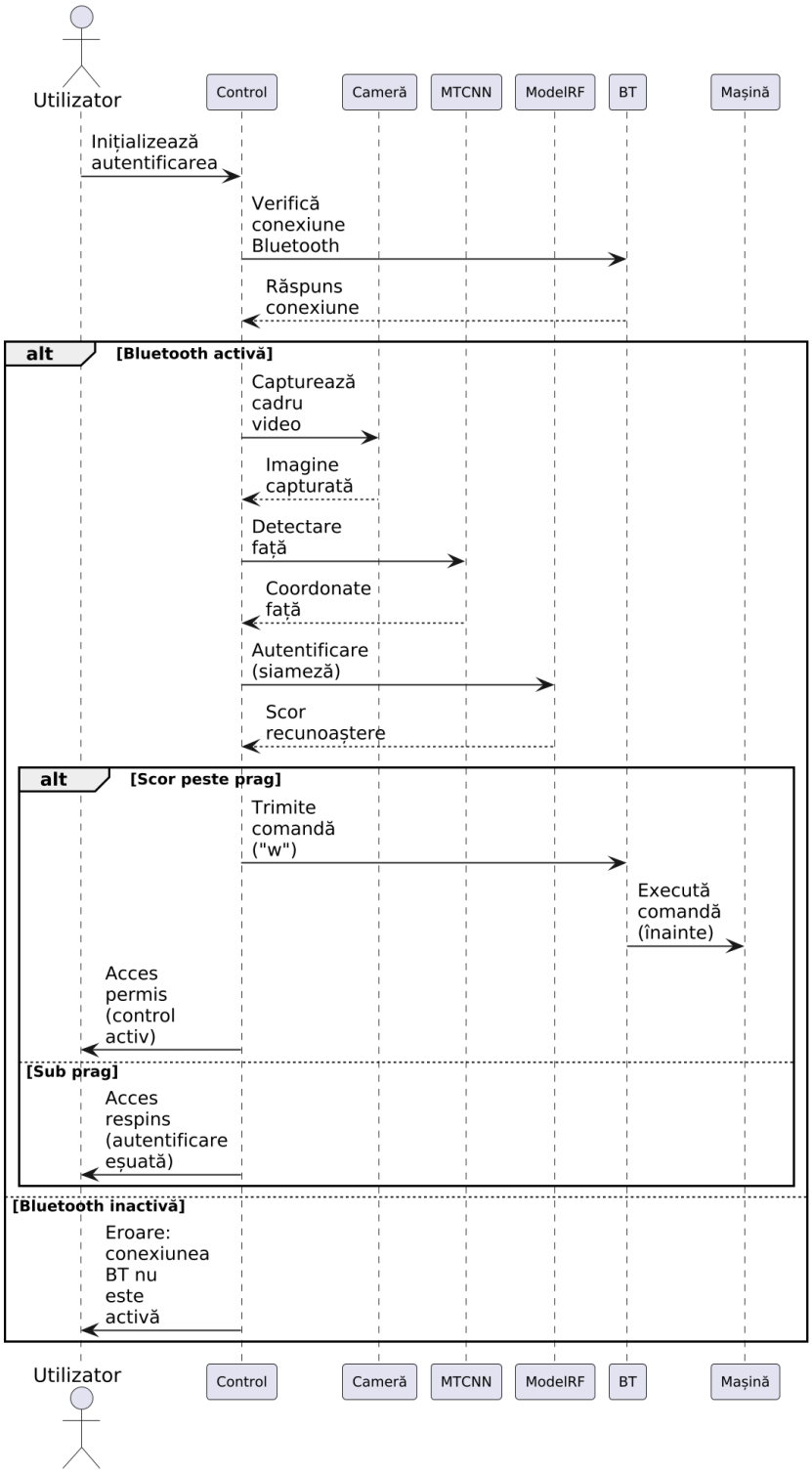


Figura 5.12: Diagrama de secvență pentru autentificarea și controlul mașinii inteligente.`https://plantuml.com/`

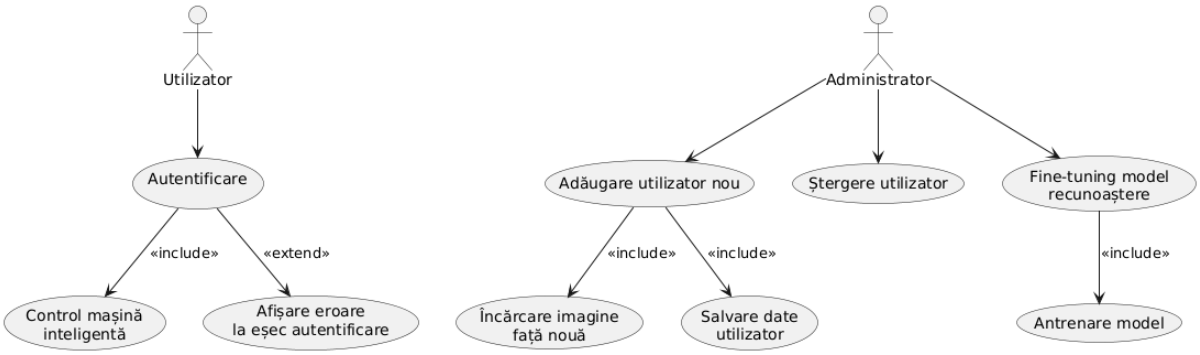


Figura 5.13: Diagrama cazurilor de utilizare pentru autentificare și administrarea utilizatorilor.`https://plantuml.com/`

Capitolul 6. Testare și validare

6.1. Componenta Machine Learning

Antrenare și Rafinare

În această parte am analizat cum s-a comportat modelul pentru validarea facială după antrenare și ajustările făcute. Am comparat prin cele mai importante metrici rezultatele obținute înainte și după fine-tuning pentru a vedea dacă a avut loc o optimizare. În contextul situației mele, ideea principală a fost să verific cât de bine funcționează modelul în practică. Cel mai important este comportamentul modelului într-un scenariu real de acces pe baza trăsăturilor faciale ale unei persoane. Evaluarea prin metrici îmi oferă validare dacă sistemul este suficient de sigur de utilizat pentru problema abordată sau nu.

Testarea modelului am realizat-o pe un set de imagini pozitive ce au în conținutul acestora, fotografii cu aceeași persoană, și pe un set de imagini negative, fotografii ale unor persoane necunoscute. Am folosit perechi pozitive și perechi negative generate pe același principiu ca în dezvoltarea modelului.

În graficul din figura (6.1) am ilustrat comportamentul modelului în cele 30 de epoci de antrenare, înainte de ajustările efectuate suplimentar. Sunt prezentate următoarele aspecte:

- Funcția de pierdere (eng. loss) la antrenare scade constant, ceea ce înseamnă că modelul își îmbunătățește performanța.
- Din cauza setului de date mic sau dezechilibrat pierderea la validare este instabilă și uneori fluctuează
- Acuratețea la antrenare crește treptat și atinge valori peste 75
- Acuratețea în timpul validării este mai instabilă, dar atinge un maxim de peste 65%, înseamnă că modelul reușește parțial să generalizeze;
- Scorul AUC (Aria reprezentată inferior curbei ROC) crește pentru antrenare și pentru validare. Ajunge la valori peste 0.75. Acest lucru arată că modelul face distincție între perechi similare și nesimilare chiar și când pragul de decizie variază.

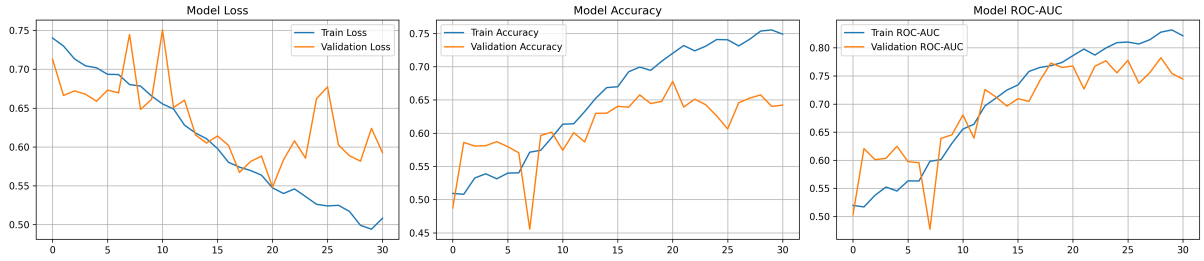


Figura 6.1: Istoricul valorilor în timpul antrenării

Instabilitatea observată din setul de validare a contribuit la alegerea mea de a adăuga un proces de îmbunătățire, motiv pentru care am aplicat ulterior ajustările descrise

în capitolele anterioare ale lucrării. Acest proces de ajustare constă în rafinarea modelului (eng. fine-tuning).

Evaluarea am împărțit-o în doua etape:

- evaluarea modelului inițial
- evaluarea modelului ajustat, dupa realizarea rafinării.

Pentru fiecare etapă în parte, am urmărit mai multe aspecte care să-mi arate cum se comportă modelul. Am folosit:

- Matricea de confuzie — pentru a vedea câte cazuri au fost clasificate corect sau greșit;
- Curba ROC și valoarea AUC — care mă ajută să înțeleg cât de bine poate modelul să facă diferența între perechi valide și invalide, indiferent de pragul de decizie;
- Alte metrici importante, printre care:
  - Acuratețea: îmi spune câte predicții au fost corecte, din total.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{6.1}$$

  - Precizia: arată cât de sigur e modelul atunci când spune că două imagini sunt ale aceleiași persoane.

$$Precision = \frac{TP}{TP + FP} \tag{6.2}$$

  - Recall-ul: măsoară cât de bine identifică toate perechile pozitive.

$$Recall = \frac{TP}{TP + FN} \tag{6.3}$$

  - Scorul F1: combină precizia și recall-ul într-o singură valoare. Mi s-a părut util mai ales când cele două valori nu sunt echilibrate.

$$F1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \tag{6.4}$$

În formulele de mai sus, coeficienții precizați și poziția din matricea de confuzie sunt:

- TP (eng. True Positive) cazurile pozitive identificate - poziția [0,0] stânga sus
- FP (False Positive) cazurile negative greșite ca pozitive - poziția [0,1] dreapta sus
- FN (False Negative) cazurile pozitive greșite ca negative - poziția [1,0] stânga jos
- TN (eng. True Negative) cazurile negative identificate - poziția [1,1] dreapta jos

Rezultate și interpretare metrici

În această subsecțiune am prezentat rezultatele obținute de model atât înainte, cât și după procesul de fine-tuning, pentru aceleași seturi de date de test. Rezultatele sunt sintetizate folosind matricea de confuzie în Figura (6.2). Modelele sunt comparate prin cei mai importanți indicatori de performanță, în Figura(6.3). Am arătat și rezultatele obținute din comparația curbelor ROC, în Figura (6.4).

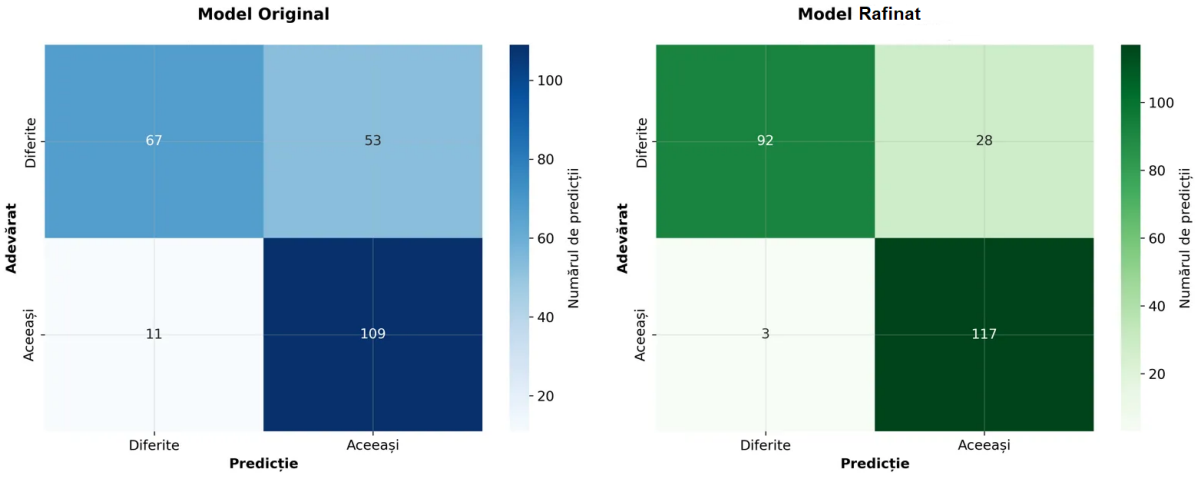


Figura 6.2: Matricea de confuzie pentru modelul original și modelul rafinat

După ce am aplicat ajustările, am observat că modelul este îmbunătățit. După rafinarea modelului, se vede diferența dintre scorurile pentru perechile pozitive (aceeași persoană) și cele negative (persoane diferite) este mai mare. Când compar două poze ale aceleiași persoane, modelul detectează mai ușor asemănarea. La fel, când are de-a face cu două persoane diferite, scorul este în descreștere și nu le mai confundă.

Acest lucru se vede și în rezultatele din matricea de confuzie: sunt cazuri mai puține în care sistemul spune că două persoane sunt aceleași, deși nu sunt.

În concluzie, modelul e mai eficient dupa ajustări. Nu mai e la fel de nesigur pe cazurile care se sunt la limită.

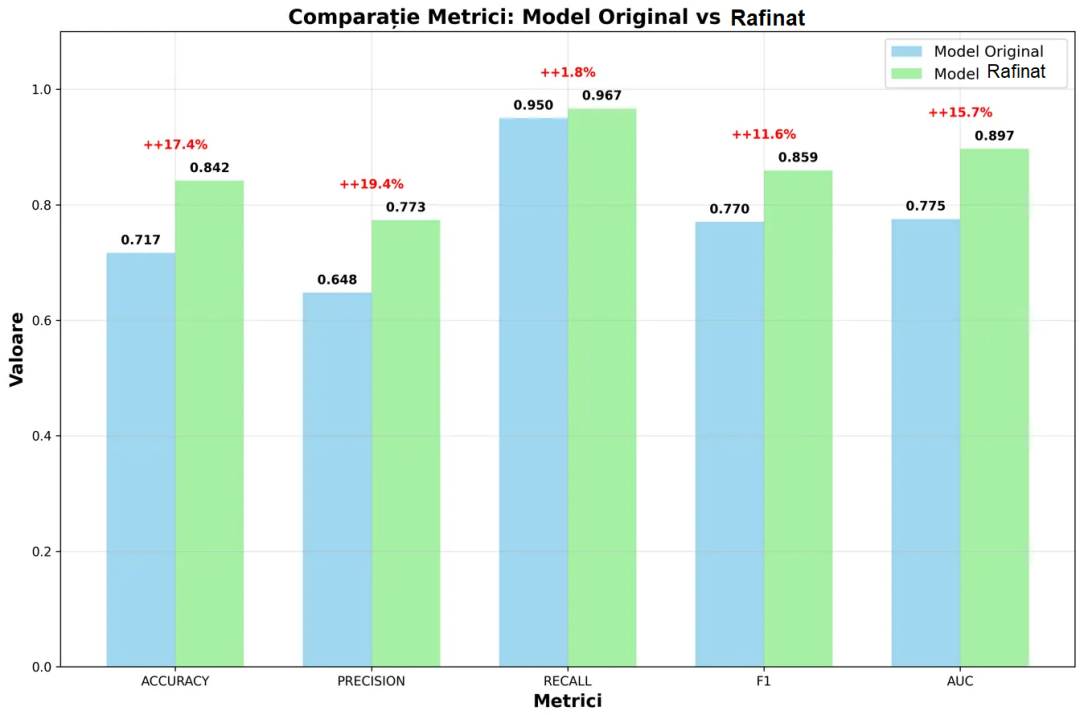


Figura 6.3: Comparația valorilor pentru metrici cheie: model original vs model rafinat

În graficul din Figura (6.3) am evidențiat cum au evoluat valorile după ce am

rafinat modelul. Fiecare secțiune din imagine arată cât de bine s-a descurcat modelul în diferite cazuri.

Se observă că toate scorurile s-au îmbunătățit. De exemplu, acuratețea a crescut. La fel și precizia, ceea ce înseamnă că atunci când modelul decide că două imagini sunt ale aceleiași persoane, greșește mai rar. Acest lucru este foarte important pentru rezolvarea problemei prezentate în lucrare.

Recall-ul a crescut și el și, practic, arată că modelul nu mai „ratează" perechi pozitive – adică nu mai zice că două poze cu aceeași persoană sunt diferite. F1 combină precizia și recall-ul, deci când ambele cresc, e clar că rezultatele sunt mai bune, în practică.

AUC-ul, spune cât de bine face diferența între perechi similare și diferite pentru orice prag și e vizibil mai mare la modelul rafinat. Cu alte cuvinte, după ajustări, modelul meu stie concret diferența între fețele care ar trebui să fie recunoscute și cele care nu.

Per total, toate aceste creșteri dovedesc că fine-tuning-ul este util, pentru a da acces numai echipajului autorizat al autovehiculului.

În Figura (6.3) sunt ilustrate valorile comparative pentru metrici cheie: acuratețe (eng. accuracy), precizie (eng. precision), recall, Scorul F1 și AUC. Fiecare metrică a adus un plus modelului:

- Acuratețea crește cu peste 17%, și ajunge la 0.842.
- Recall-ul depășește valoarea de 0.96, indicând o rată foarte bună de recunoaștere a perechilor pozitive.
- Precizia este în creștere cu aproape 20%, ceea ce înseamnă că numărul de alarme false scade.

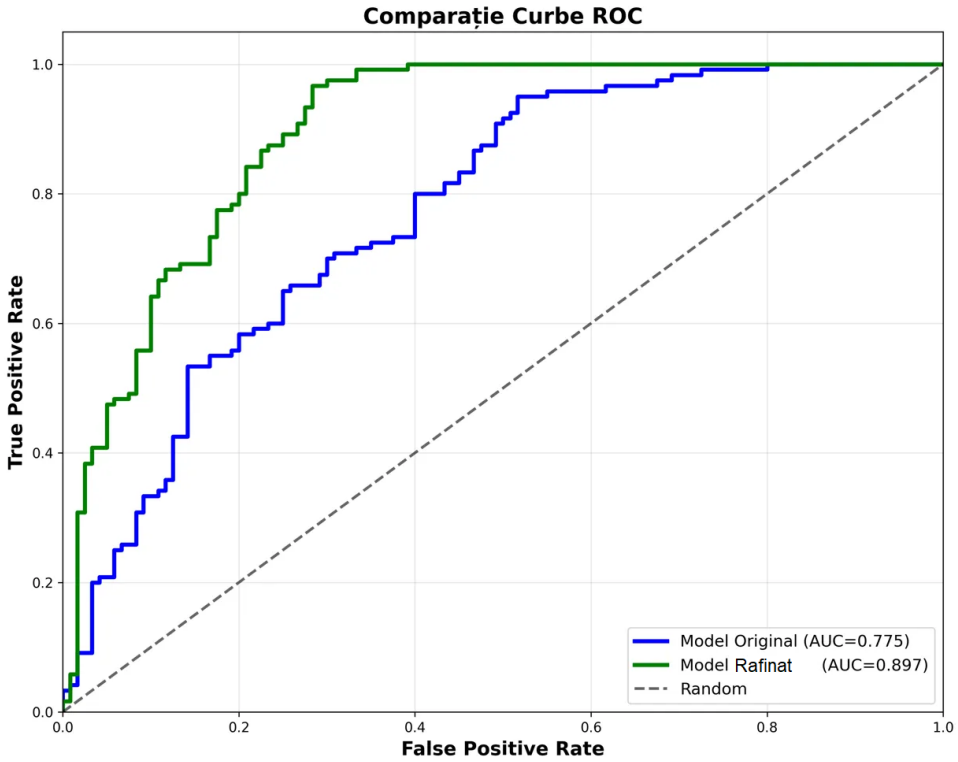


Figura 6.4: Compararea curbelor ROC pentru modelul original și modelul rafinat

Diferenţa dintre cele două modele se vede în curbele ROC din figura 6.4. Modelul îmbunătăţit, l-am reprezentat cu verde și are o suprafaţă de 0.897. Modelul original are valoarea de 0.775 și este mai mică decât modelul rafinat. Modelul rafinat separă mult mai bine exemplele corecte de cele incorecte, indiferent de pragul pe care l-am ales.

Prin ajustările aduse, modelul de recunoaştere facială a devenit mai bun în situația unui sistem de verificare facială. Într-un context real de securitate, modelul rafinat este specializat pe imaginile persoanelor cunoscute. Datorită rafinării, modelul nu mai greșește ca înainte.

Testarea augmentării datelor

Pentru testare, am creat imagini augmentate să mă asigur că modificările sunt generate corect. Pentru fiecare fotografie originală am aplicat niște augmentări aleatorii, am variat luminozitatea, contrastul și saturația, am convertit unele cadre în tonuri de gri, și am făcut pe axa orizontală o rotire. Imaginile augmentate le-am folosit ca intrare în rețea pentru a adăuga varietate datelor.

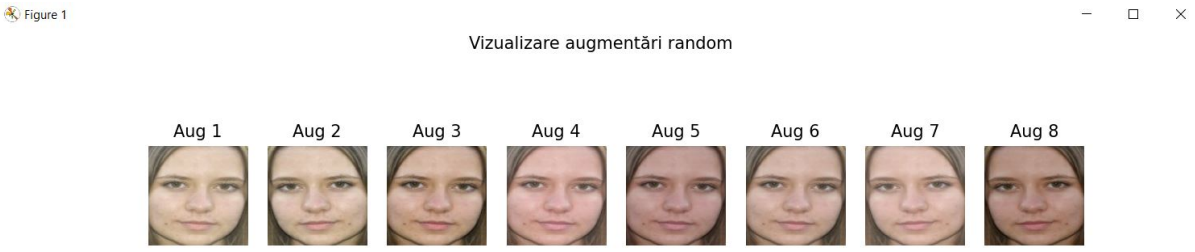


Figura 6.5: Vizualizare rezultat augumentare de date

6.2. Componenta Hardware

În procesul de testare a părții hardware, am combinat observații de tip calitativ și elemente cantitative. Calitatea componentei este expusă prin modul în care sistemul se comportă în diferite situații reale. Analiza cantitativă este reprezentată prin măsurători aproximative asemenea distanțelor detectate de senzori sau timpul de răspuns la comenzi. Chiar și în cazul în care nu s-au desfășurat teste într-un mediu industrial sau într-un vehicul real, am urmat logica componentelor principale unui autovehicul și modul în care componentele lucrează între ele.

Validarea efectivă a părții hardware s-a bazat pe o înțelegere funcțională a sistemului. Am verificat dacă legăturile dintre senzori, motor, microcontroler și modulul Bluetooth sunt conforme și dacă sistemul reacționează coerent când primește date. În lipsa unui mediu 100% real, modul în care răspund componentele oferă o viziune că hardware-ul este pregătit pentru situații reale, cel puțin din punct de vedere al logicii și al funcționării interne.

Verificarea configurației hardware

Înainte de rularea oricărui test, am parcurs următorii pași pentru fiecare componentă hardware utilizată (STM32F303RE, HC-SR04, HC-05, motoare DC şi driver L298N):

- Am verificat conexiunile electrice pentru a mă asigura că fiecare pin este legat corect.
- Am confirmat compatibilitatea tensiunilor de alimentare cu specificaţiile fiecărui modul.
- În STM32CubeMX, am alocat şi remapat pinii necesari fiecărui periferic.
- Am testat funcţionarea logică a codului, interacţionând direct cu fiecare componentă.

Exemple de configurări concrete:

- HC-SR04: Senzorii ultrasonici pentru spate şi faţă au fost conectaţi la timerele interne TIM1, respectiv TIM3. Semnalele TRIG şi ECHO sunt măsurate prin aceste timere pentru a determina distanţa.
- HC-05: Modulul Bluetooth a fost configurat pe interfaţa USART1, remapată pe pinii PC4 (TX) şi PC5 (RX), asigurând comunicarea stabilă cu aplicaţia de pe laptop.
- Motoare DC: Am generat semnale PWM cu ajutorul TIM2, care controlează puntea H din driverul L298N pentru a seta viteza şi direcţia motoarelor.

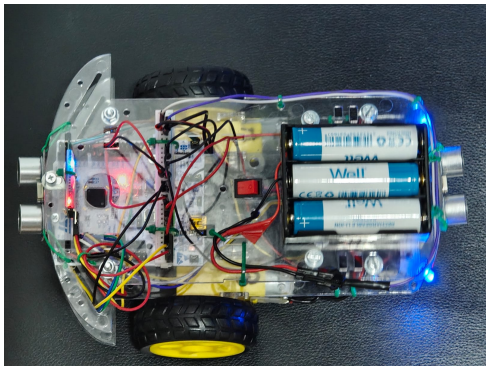


Figura 6.6: Conexiuni Robot - Vedere de sus

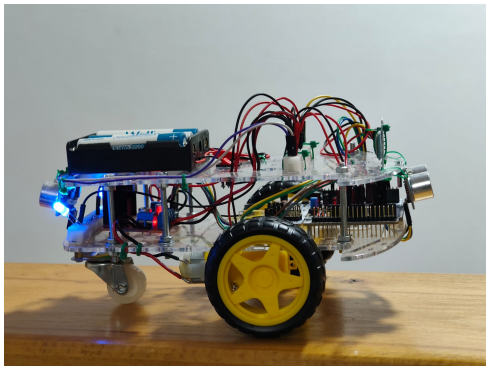


Figura 6.7: Conexiuni Robot - Vedere din lateral

În figurile 6.6 şi 6.7 se văd ansamblul de componentelor şi cablajul robotului, pentru testare. După montarea lor am parcurs cu multimetru pentru a vedea tensiunile și dacă sunt ceva întreruperi. Am verificat tot robotul și nu am observat nicio problemă sau scurtcircuit. Sistemul este pregătit pentru conectarea cu aplicația software.

Scenarii de testare funcțională

Au fost definite mai multe scenarii reprezentative de testare logică:

- Obstacol detectat sub 10 cm (față/spate): Sistemul dezactivează automat semnalul PWM, oprind mișcarea robotului.
- Îndepărtarea obstacolului: După ce distanța față de senzor revine peste limita setată, PWM este reactivat, iar robotul își reia mișcarea.
- Transmiterea comenzilor prin Bluetooth: Comenzile „față", „stop" și „spate" transmise prin aplicația grafică (PyQt5) sunt recepționate prin modulul HC-05 și interpretate corect de microcontroler.

Capitolul 7. Manual de instalare și utilizare

7.1. Cerințe pentru instalare

Pentru a rula aplicația, sunt necesare următoarele resurse hardware și software:

Cerințe hardware

- Procesor: minim Intel i5 sau echivalent AMD
- Memorie RAM: minim 8 GB (recomandat 16 GB)
- Spațiu liber pe disc: minim 5 GB
- Cameră web: necesară pentru capturarea imaginilor în timp real
- Port USB sau Serial: pentru conectarea cu eventuale componente hardware externe

Cerințe software

- Sistem de operare: Windows 10/11, macOS 10.15+, sau Ubuntu 18.04+
- Python: versiunea 3.8 – 3.10 (nu este compatibil cu versiuni mai noi)
- Conexiune la internet: necesară pentru descărcarea pachetelor

7.2. Instalarea aplicației

Instalarea Python

Limbajul Python se poate instala de la adresa oficială: https://www.python.org/downloads/, sub formă de kit. Este necesar un kit în versiunea 3.8, 3.9, 3.10, pentru rularea corectă. Versiunile mai noi decât 3.11 nu sunt compatibile.

Crearea mediului virtual

În continuare, se deschide linia de comandă și se navighează în directorul `Model-Siamese` din structura proiectului. În acest director se creează un mediu virtual și se activează folosind comenzile:

Pentru sistemul de operare Windows:

```
python -m venv venv
venv\Scripts\activate
```

Pentru macOS sau Linux:

```
python -m venv venv
source venv/bin/activate
```

7.2.1. Instalarea bibliotecilor necesare

După activarea mediului virtual, se instalează bibliotecile necesare folosind următoarele comenzi:

```
pip install -upgrade pip
pip install tensorflow==2.13.0
pip install opencv-python
pip install PyQt5
pip install mtcnn
pip install numpy
pip install pillow
pip install pyserial
```

7.2.2. Finalizarea instalării

Aplicația se va executa din directorul APLICATIE, folosind comanda: `python main.py` Executarea acestei comenzi va deschide interfața grafică a aplicației.

7.3. Utilizarea aplicației

Interfața aplicației este organizată pe mai multe ferestre. La pornirea aplicației, se deschide fereastra principală, în care utilizatorul este autentificat prin recunoaștere facială. Dacă fața este recunoscută, apare mesajul „Acces permis", altfel apare „Acces respins". În partea dreaptă se află panoul de control al robotului și opțiunea de înregistrare utilizator:

- Butonul `START` pornește verificarea facială, după validare oferă acces la tastele pentru direcții.
- Butonul `STOP` oprește comunicarea și controlul.
- Reprezentarea vizuală a tastelor (controlul se face de la tastatură): `w` (față), `s` (spate), `a` (stânga), `d` (dreapta), `b` (stop motoare).
- Se introduce portul Bluetooth și se apăsa butonul „ConectareBT" pentru conectare.

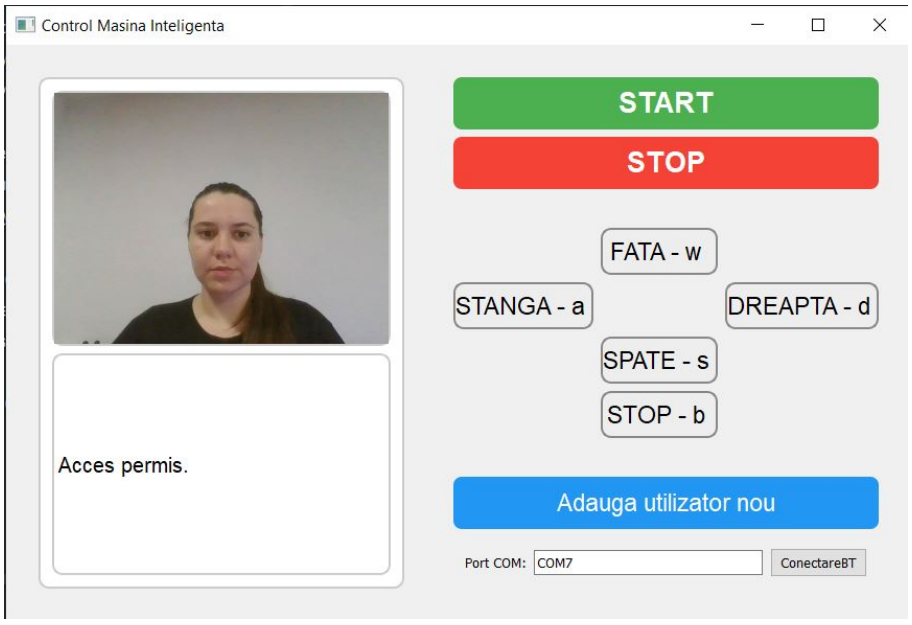


Figura 7.1: Fereastra principală a aplicației

7.3.1. Conectarea aplicației prin Bluetooth

Pentru conectarea aplicației la robot este necesar să se facă o conectare inițială între laptop și modulul Bluetooth. Acest proces se face o singură dată și presupune următorii pași:

1. Se pornește modulul Bluetooth al mașinuței.
2. Pe laptop, se accesează meniul de setări Bluetooth, se caută dispozitivele disponibile și se selectează dispozitivul numit `HC-05-licenta`.
3. La solicitarea parolei, se introduce codul „2025” (conform configurării modulului din Capitolul 5, subsecțiunea „Configurarea modulului Bluetooth HC-05”).

După ce dispozitivul este împerecheat (pairing complet), trebuie să se identifice portul serial (COM) asociat acestuia.

Portul identificat (ex. COM7, COM10 etc.) va fi introdus în câmpul „Port COM” din interfața aplicației la fiecare pornire, iar apoi se apasă butonul `ConectareBT`.

7.3.2. Înregistrarea unui utilizator nou

Pentru a adăuga o persoană nouă în sistem, utilizatorul apasă butonul „Adaugă utilizator nou”, butonul din Figura (7.1). Pentru securitate se va introduce un cod cunoscut numai de către o persoană autorizată. Trecerea la pasul următor se face numai după ce a fost acționat butonul „Verifică parola” din Figura (7.2).

Figura 7.2: Fereastra de introducere parolă pentru adăugare utilizator nou

Parola implicită pentru această operațiune este „1234”.

Figura 7.3: Format utilizatori

După validarea parolei, se va trece la înregistrarea unei noi persoane în sistem. Fiecare utilizator este reprezentat printr-un rând care include numele și trei acțiuni posibile:

- `Cropped`: permite vizualizarea imaginilor procesate automat (doar porțiunea cu fața detectată); 7.5
- `Original`: deschide galeria cu imaginile originale capturate pentru acea persoană; 7.4
- `Elimină persoana`: șterge complet utilizatorul și toate imaginile asociate acestuia.

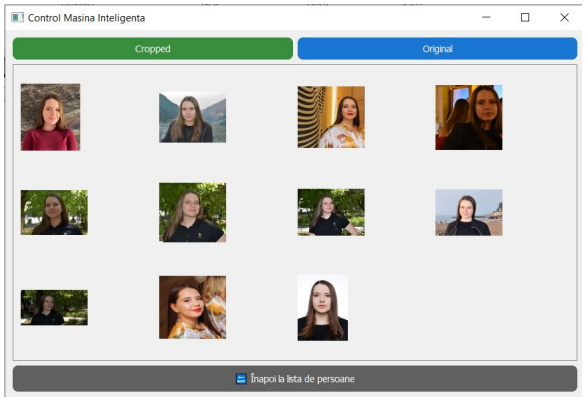


Figura 7.4: Imagini originale ale utilizatorului

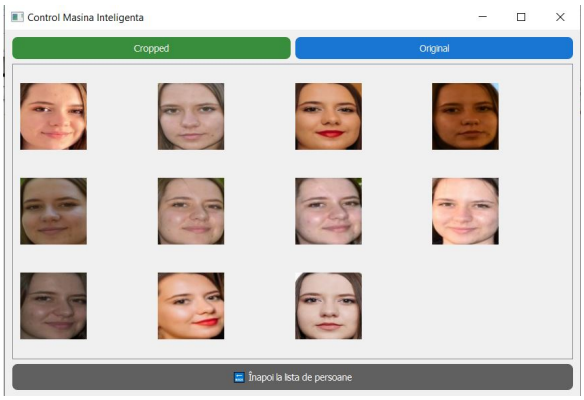


Figura 7.5: Imagini procesate (cropped)

În partea de jos a ferestrei sunt disponibile următoarele funcționalități:

- `Adaugă persoană nouă`: permite înregistrarea unui nou utilizator (va relua procesul de introducere a numelui și captură de imagini);
- `Înapoi la ecranul principal`: revine la fereastra principală;
- `Adaugă în sistem (fine-tune model cu toate persoanele)`: antrenează din nou modelul de recunoaștere facială, folosind toți utilizatorii și imaginile existente. Această funcționalitate este utilă după ce s-au adăugat sau eliminat utilizatori și permite îmbunătățirea performanței sistemului.

7.4. Instrucțiuni suplimentare

- Se recomandă iluminare uniformă pentru rezultate bune.
- Este indicat să se capteze imagini din unghiuri ușor diferite la înregistrare.

Capitolul 8. Concluzii

În această lucrare am demonstrat prin simularea unui autovehicul printr-un robot, că vehiculul poate hotărî singur cui permite să îl controleze. Toate acestea fără chei, carduri sau conexiune la internet.

8.1. Rezumatul contribuțiilor în raport cu cerințele funcționale

Concret, am realizat următoarele:

- Proiectarea şi asamblarea robotului
  Am montat robotul, fixând motoarele şi senzorii pe şasiu, cositorind cablurile pentru motoare și am conectat pe placa STM32 modulele, apoi am asamblat şi alimentând întregul ansamblu. În secțiunea 5.2. Componenta hardware, din capitolul 5, se pot vedea conexiunile făcute.

- Firmware STM32 pentru controlul modulelor
  Logica de pornireoprire a motoarelor, viraje, procesul de evitare a obstacolelor dar și comunicația serială este explicată în Capitolul 5, din subsecțiunea: „Integrarea componentelor hardware și logica de control firmware".

- Aplicaţia pentru recunoaştere facială
  Am dezvoltat o aplicaţie care preia imaginea video de la cameră, găseşte automat faţa din cadru, o decupează, calculează semnătura numerică şi o compară cu utilizatorii înregistrați. În aceeaşi interfaţă pot înscrie sau şterge profiluri. Detalii complete se găsesc în Capitolul 5, Secțiunea 5.3.

- Interfaţa de administrare a utilizatorilor
  Am creat o fereastră unde administratorul vede lista de utilizatori, adaugă fotografii noi, şterge profiluri. Administratorul este obligat să introducă un cod pentru a efectua aceste operații asupra datelor din sistem. Toate acestea se efectuează fără oprirea aplicaţiei sau a robotului. Detalii complete le-am expus în Capitolul 5, Secțiunea 5.3.

- Colectarea dataset-ului şi antrenarea modelului
  Am pornit de la un set public de imagini cu fețele mai multor persoane diferite. Am selectat şi procesat minim 90 de fotografii pentru fiecare utilizator. Ulterior am făcut redimensionare, augmentare şi normalizare, aşa cum am explicat pas cu pas în Capitolul 4, Secțiunea 4.2, înainte de a antrena modelul care rulează local.

- Testarea sistemului în scenarii reale
  Am supus sistemul la încercări în situaţii diferite. Am testat verificarea facială într-o lumină foarte slabă, dar și în diferite unghiuri. Am verificat toate funcțiile de mișcare ale robotului. Toate rezultatele sunt detaliate în Capitolul 6, Secțiunea 6.2.

8.2. Analiză critică a rezultatelor din cerințele non-funcționale

- Precizie.
  În condiţii de iluminare obişnuită, sistemul validează corect utilizatorii autorizaţi şi respinge majoritatea acceselor nepermise. Sub un anumit prag de lumină, rata erorilor creşte, din această cauză este nevoie de o lumină minimă la bord.

- Timp de răspuns.
  Intervalul dintre detecţia feţei şi comanda de pornire a motorului se menţine, în medie, sub o secundă. Astfel am reușit să încadrez timpul de răspuns în cerinţele unei intervenţii rapide.

- Operare fără internet.
  Testele efectuate cu conexiunea la internet dezactivată au confirmat că înregistrarea, recunoaşterea şi conducerea automobilului se realizează local.

- Scalabilitate.
  Adăugarea multor profiluri creează o mică întârziere suplimentară, iar consumul de memorie al computerului de bord creşte vizibil. Dacă personalul echipajului se extinde, va fi nevoie de resurse RAM sau de optimizarea modului în care sunt gestionate datele.

8.3. Direcţii de dezvoltare

- Implementare pe autospecială reală
  Următorul pas important este să implementez sistemul pe o maşină de intervenţie adevărată, legând modulul de recunoaştere la contactul motorului şi la reţeaua CAN (eng. Controller Area Network) a vehiculului.

- Creşterea preciziei modelului
  Vreau să colectez mai multe imagini în situaţii reale cu echipament complet, lumină slabă. Vreau să perfecţionez algoritmul, astfel încât recunoaşterea să fie şi mai sigură, inclusiv în condiţii dificile.

- Criptarea bazei de date şi securizarea ei
  Ca implementare viitoare, vreau să criptez baza de imagini şi vectorii stocaţi pe computerul de bord prin parole sau chei hardware. Datele echipajului trebuie să rămână protejate chiar dacă dispozitivul sistemului este compromis.

8.4. Încheiere

Prin rezultatele prezentate am dovedit că obiectivul de bază, pornirea vehiculului doar de către persoane autorizate, independent de internet, a fost atins. Am transformat cerinţele stabilite la început în soluţii concrete de hardware şi software, le-am testat în scenarii realiste şi am confirmat performanţele de precizie şi operare offline. Prototipul demonstrează că verificarea facială poate fi o metodă de securizare pentru autospecialele de intervenţie, deschizând drumul către integrarea pe un vehicul real.

Bibliografie

[1] Neurotechnology, "Verilook sdk – biometric facial identification technology," https://www.neurotechnology.com/verilook.html, 2024, accesat în iulie 2025.

[2] D. Grigorescu and A. M. Mihăescu, "Comparative analysis of face recognition sdks for embedded systems," in *Proceedings of the IEEE International Conference on e-Health and Bioengineering (EHB)*, 2021, pp. 1–4.

[3] Regula Forensics, "Face sdk– face matching and liveness detection," https://regulaforensics.com/products/face-recognition-sdk/, 2024, accesat iulie 2025.

[4] J. R. McConvey, "Regula releases update to face sdk with improved liveness detection," *BiometricUpdate*, octombrie 2024.

[5] Kairos, "Face recognition api reference docs," https://face.kairos.com/docs/api, 2025, accesat iulie 2025.

[6] BiometricUpdate, "Biometric liveness detection platform launched by kairos," *BiometricUpdate*, mai 2024.

[7] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition," in *ICML Deep Learning Workshop*, 2015. [Online]. Available: https://www.cs.cmu.edu/~rsalakhu/papers/oneshot1.pdf

[8] Y. Guo and L. Zhang, "One-shot face recognition by promoting underrepresented classes," *arXiv preprint arXiv:1707.05574*, 2017. [Online]. Available: https://www.microsoft.com/en-us/research/wp-content/uploads/2017/07/one-shot-face.pdf

[9] Z. Ding, Y. Guo, L. Zhang, and Y. Fu, "Generative one-shot face recognition," *arXiv preprint arXiv:1910.04860*, 2019. [Online]. Available: https://arxiv.org/abs/1910.04860

[10] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 815–823. [Online]. Available: https://arxiv.org/abs/1503.03832

[11] O. Arasi, "Face recognition system using facenet: A review," *ResearchGate*, 2023. [Online]. Available: https://www.researchgate.net/publication/376516541_Face_Recognition_System_Using_FaceNet_A_Review

[12] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 4690–4699. [Online]. Available: https://openaccess.thecvf.com/content_CVPR_2019/papers/Deng_ArcFace_Additive_Angular_Margin_Loss_for_Deep_Face_Recognition_CVPR_2019_paper.pdf

[13] Y. Srivastava, V. Murali, and S. R. Dubey, "A performance comparison of loss functions for deep face recognition," *arXiv preprint arXiv:1901.05903*, 2019. [Online]. Available: https://arxiv.org/abs/1901.05903

[14] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "Sub-center arcface: Boosting face recognition by large-scale noisy web faces," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. [Online]. Available: https://www.ecva.net/papers/eccv_2020/papers_ECCV/papers/123560715.pdf

[15] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1701–1708, 2014. [Online]. Available: https://doi.org/10.1109/CVPR.2014.220

[16] STMicroelectronics, "Stm32f303re datasheet," https://www.st.com/resource/en/datasheet/stm32f303re.pdf, 2019, accessed: June 2025.

[17] K. Kassem, "Stm32 hal library tutorial and examples," https://deepbluembedded.com/stm32-hal-library-tutorial-examples/, n.d., accessed: 2025-06-22.

[18] ITead Studio, "Hc-05 bluetooth module – datasheet," https://components101.com/sites/default/files/component_datasheet/HC-05%20Datasheet.pdf, 2010, accessed: June 2025.

[19] SparkFun Electronics, "Hc-sr04 ultrasonic sensor datasheet," http://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf, n.d., accessed: June 2025.

[20] Sensor Partners, "Everything about the operation principles of ultrasonic sensors," https://sensorpartners.com/en/knowledge-base/everything-about-the-operation-principles-of-ultrasonic-sensors/, 2023, accesat în iunie 2025.

[21] N. Semiconductor, "Pulse width modulation (pwm) - nrf connect sdk intermediate," https://academy.nordicsemi.com/courses/nrf-connect-sdk-intermediate/lessons/lesson-4-pulse-width-modulation-pwm/topic/pulse-width-modulation-pwm/, 2024, accesat în iunie 2025.

[22] Øyvind Nydal Dahl, "How an h-bridge works – build electronic circuits," 2021, accesat în iunie 2025. [Online]. Available: https://www.build-electronic-circuits.com/h-bridge/

[23] "L298n dual h-bridge motor driver module," https://www.sparkfun.com/datasheets/Robotics/L298_H_Bridge.pdf, accessed: 2025-06-22.

[24] X. Deng *et al.*, "H-bridge l298n," https://www.researchgate.net/figure/H-BridgeL289N-Source6_fig6_351660996, 2021, accessed: 2025-06-22.

Anexa A. Alte informații relevante (demonstrații etc.)

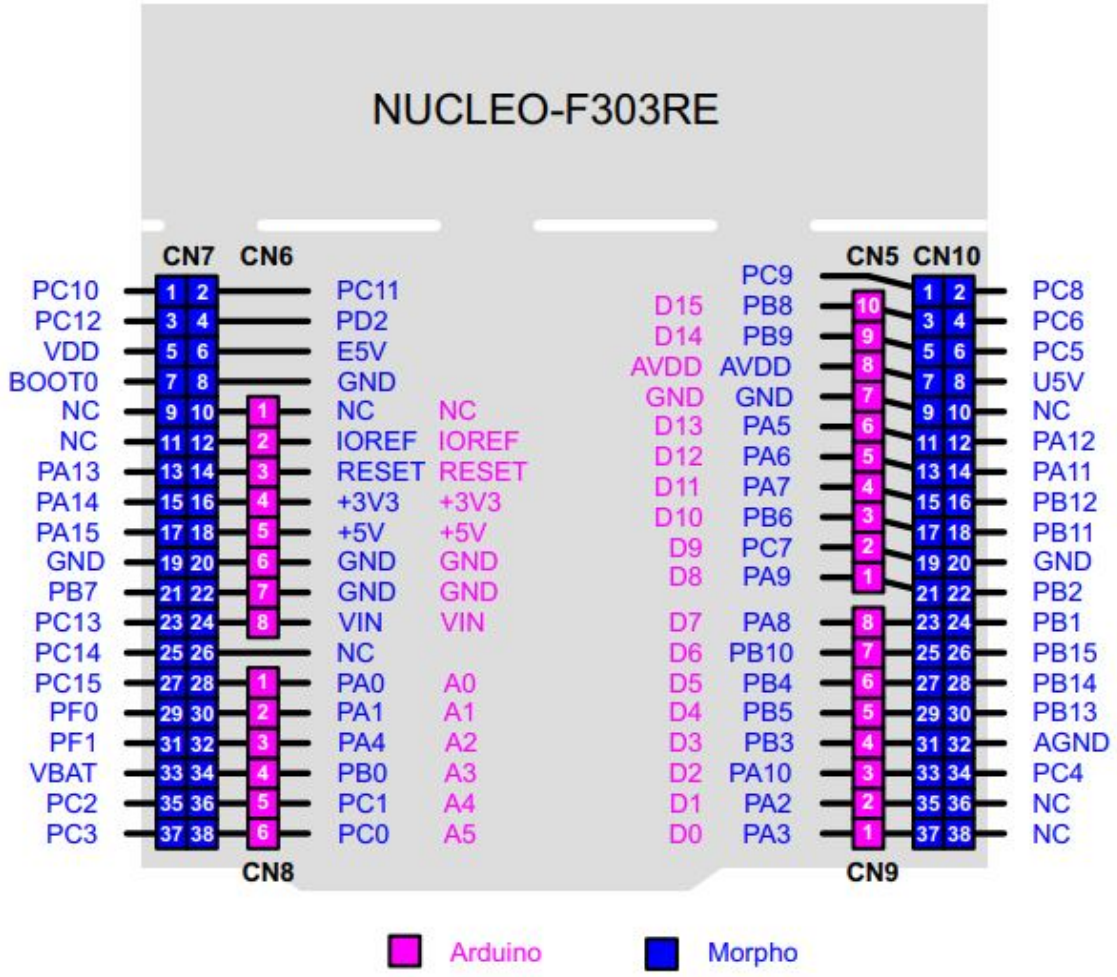


Figura A.1: Distribuția completă a pinilor pentru placa NUCLEO-F303RE[1]

[1]Distribuția pinilor este preluată din documentația oficială STMicroelectronics – Secțiunea 6.10 „Extension connectors". Sursa: `https://www.st.com/resource/en/user_manual/um1724-stm32-nucleo64-boards-mb1136-stmicroelectronics.pdf`

Table 7. External power sources

Input power name	Connectors pins	Voltage range	Max current	Limitation
VIN	CN6 pin 8 CN7 pin 24	7 V to 12 V	800 mA	From 7 V to 12 V only and input current capability is linked to input voltage: 800 mA input current when Vin = 7 V 450 mA input current when 7 V < Vin <= 9 V 250 mA input current when 9 V < Vin <= 12 V
E5V	CN7 pin 6	4.75 V to 5.25 V	500 mA	-

Table 8. Power-related jumper

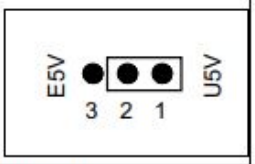
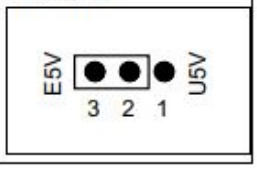
Jumper	Description
JP5	U5V (ST-LINK VBUS) is used as a power source when JP5 is set as shown below (Default setting) 
	VIN or E5V is used as a power source when JP5 is set as shown below. 

Figura A.2: Poziționarea jumperului JP5 pentru selectarea sursei de alimentare[2]

[2]Schema preluată din documentația oficială STMicroelectronics – Capitolul 5.1 „Power supply configuration". Sursa: `https://www.st.com/resource/en/user_manual/um1724-stm32-nucleo64-boards-mb1136-stmicroelectronics.pdf`