Inginerie electron., telecomunicaţii și tehnologii informaționale (20.20.100) - Electronică aplicată (20.20.100.10)
2022

SISTEM VIDEO PENTRU DETECȚIA URMĂRITORILOR DIN TRAFIC

Candidat: Nastasia-Elena, Nițu

Coordonator științific: Asist. SL dr.ing. ec. Valentin-Adrian, Niță

Sesiunea: Iunie 2022

REZUMAT

Acestă lucrare de diplomă este realizată în cadrul departamentului de Electronică Aplicată al Universității Politehnica din Timișoara.

Recunoașterea Automată a Numerelor de Înmatriculare integrează tehnici de Computer Vision (și nu numai) în cadrul domeniului Sistemelor Inteligente de Transport, domeniu din ce în ce mai important în ultimii ani.

Domeniul Recunoașterii Automate a Numerelor de Înmatriculare din trafic nu este dezvoltat în totalitate pentru România față de dezvoltările existente în alte țări. Creșterea cercetărilor în cadrul acestui domeniu reprezintă un factor pozitiv pentru securitatea în trafic și nu numai. Dezvoltarea unor instrumente specializate pentru România prezintă o importanță maximă, dacă nu chiar o necesitate.

Sistemul propus în cadrul acestui proiect preia ca input flux de date în timp real de folosind o cameră web conectată unui sistem Raspberry Pi, procesează fluxul de date folosind tehnologii de actualitate și returnează atenționări instantanee cu privire la posibile autoturisme ce pot urmări utilizatorul în trafic în scopuri diverse.

Lucrarea folosește conceptele de procesare de imagine și învățare automată pentru a duce la final sarcinile impuse.

Testarea sistemului în timp real dovedește că metoda propusă oferă rezultatele așteptate, cu o acuratețe mulțumitoare. Rezultatele deschid o cale către dezvoltări ulterioare, ce pot fi componente importante în cadrul domeniilor Automotive, Sisteme Inteligente de Transport și Securitate Personală.

Cuvinte cheie: Python, Raspberry Pi, OCR, Procesare de imagine, Învățare prin transfer, Tensorflow

ABSTRACT

This diploma thesis is carried out within the department of Applied Electronics of the Polytechnic University of Timișoara.

Automatic License Plate Recognition integrates Computer Vision techniques (and more) into the field of Intelligent Transportation Systems, a field that has become increasingly important in recent years.

The field of Automatic License Plate Recognition is not fully developed for Romania compared to existing developments in other countries. Increasing research in this area is a positive factor for traffic safety and beyond. The development of specialized tools for Romania is of utmost importance, if not a necessity.

The system proposed in this project takes as input real-time data flow using a webcam connected to a Raspberry Pi system, processes the data flow using current technologies and returns instant alerts on possible cars that could be following the user in traffic in various purposes.

The paper uses the concepts of Image Processing and Machine Learning to complete the required tasks.

Real-time system testing proves that the proposed method delivers the expected results with satisfying accuracy. The results pave the way for further developments, which could be important components in the fields of Automotive, Intelligent Transportation Systems and Personal Security.

Keywords: Python, Raspberry Pi, OCR, Image Processing, Transfer Learning, Tensorflow

Mulțumiri

Rămân profund recunoscătoare domnului Valentin-Adrian Niță, pentru timpul, expertiza, răbdarea, sfaturile și inspirația oferite pe tot parcursul realizării acestui proiect. Fără sprijinul acestuia, acest proiect nu ar fi putut exista.

Domnului Dan Lascu îi mulțumesc pentru că a fost mai mult decât un decan, ascultând cu adevărat doleanțele fiecărui student, dovedindu-și empatia și compasiunea în fiecare zi.

Nu îi pot uita nici pe fratele meu Cătălin, pe părinții mei, pe bunicii mei, pe Alexandru, pe doamna profesoară Ana Borcean, pe domnul profesor Mircea-Petru Rusu și pe psih. Doru Constantin Bălan, care mi-au oferit sprijinul și m-au îndrumat mereu către calea ce îmi era potrivită. Nu aș fi ajuns aici fără voi.

Listă figuri

Listă tabele

Cuprins

1. INTRODUCERE

Recunoașterea automată a plăcuțelor de înmatriculare (eng. Automatic Number Plate Recognition – ANPR) este tehnologia ce utilizează recunoașterea optică a caracterelor (eng. Optical Character Recognition – OCR) pe fișiere de tip imagine pentru a citi caracterele alfanumerice de pe plăcuțele de înmatriculare a vehiculelor, cu scopul, de obicei, de a capta date despre vehicul și locația acestuia [1].

Alte denumiri folosite frecvent:

- Automatic (automated) license-plate recognition (ALPR),
- Automatic (automated) license-plate reader (ALPR),
- Automatic vehicle identification (AVI),
- Automatisk nummerpladegenkendelse (ANPG),
- Car-plate recognition (CPR),
- License-plate recognition (LPR),
- Lecture automatique de plaques d'immatriculation (LAPI),
- Mobile license-plate reader (MLPR),
- Vehicle license-plate recognition (VLPR),
- Vehicle recognition identification (VRI),

În continuare, vom folosi majoritar termenul de ANPR, cu anumite abateri în citări.

ANPR poate folosi televiziunea cu circuit închis (eng. Closed-Circuit Television – CCTV) și camere de supraveghere cu scopul de a monitoriza respectarea regulilor rutiere sau camere special concepute pentru această sarcină [1].

Această tehnologie este folosită majoritar de forțele de poliție din întreaga lume în scopuri de aplicare a legii (respectarea limitelor de viteză legale sau determinarea cazului în care un vehicul este înmatriculat sau autorizat). De asemenea, tehnologia este folosită pentru colectarea electronică a taxelor pe drumurile ce necesită plata unei taxe pentru utilizare, dar și ca metodă de catalogare a fluxului de trafic, de exemplu de către agențiile publice de autostrăzi.

Recunoașterea automată a plăcuțelor de înmatriculare poate fi folosită pentru a stoca imaginile surprinse de camere, precum și textul de pe plăcuța de înmatriculare, dar și alte aspecte (unele sisteme sunt configurate pentru a stoca și o fotografie a șoferului).

Sistemele folosesc în mod obișnuit tehnologia cu infraroșu pentru a permite camerei să facă poze la orice oră din zi sau din noapte [2] [3].

ANPR a fost inventat în 1976 la sediul Police Scientific Development Branch din Marea Britanie. Sistemele prototip funcționau deja în 1979 și au fost realizate contracte

pentru producerea de sisteme industriale, mai întâi în colaborare cu EMI Electronics, iar apoi Computer Recognition Systems (CRS, acum parte a companiei Jenoptik) din Wokingham, Marea Britanie. Sistemele prototip timpurii au fost implementate pe drumul A1 din Regatul Unit și în tunelul Dartford din aceeași națiune. Prima arestare prin detectarea unei mașini furate a fost făcută în 1981. Tehnologia ANPR, însă, nu a devenit utilizată pe scară largă decât abia în anii 1990, când au fost lansate noi dezvoltări în software mai ieftine și mai ușor de utilizat [4].

Colectarea datelor ANPR pentru utilizare ulterioară (adică pentru soluționarea infracțiunilor neidentificate la momentul realizării acestora) a fost documentată la începutul anilor 2000. Primul caz documentat în care ANPR a fost folosit pentru a ajuta la rezolvarea unei crime a avut loc în noiembrie 2005, în Bradford, Marea Britanie, la condamnarea ucigașilor lui Sharon Beshenivsky, agent de Poliție britanic [5].

Aspectul software al sistemului rulează pe hardware-ul standard al unui computer și poate fi conectat la alte aplicații sau baze de date. Mai întâi se utilizează o serie de tehnici de procesare de imagine pentru a detecta, normaliza și îmbunătăți imaginea plăcuței de înmatriculare, iar apoi se realizează recunoașterea optică a caracterelor pentru a extrage caracterele alfanumerice ale plăcuței de înmatriculare.

Sistemele ANPR sunt, în general, implementate într-una dintre cele două abordări de bază: una permite ca întregul proces să fie efectuat la locația preluării de imagine în timp real, iar celălalt transmite toate imaginile la un computer aflat la distanță și realizează, ulterior, procesarea.

Când recunoașterea automată a numărului de înmatriculare se realizează la locul de bază, informațiile capturate ale plăcii alfanumerice, data, ora și orice alte informații necesare sunt procesate complet în aproximativ 250 de milisecunde [1]. În celălalt aranjament, există de obicei un număr mare de PC-uri interconectate pentru a gestiona sarcini mari de lucru. Adesea, în astfel de sisteme, există o cerință de a transmite imagini către serverul aflat la distanță, iar acest lucru poate necesita medii de transmisie cu lățime de bandă mare.

Utilizări:

- Aplicarea legilor rutiere (identificarea vehiculelor neînmatriculate, furate și neasigurate, precum și a șoferilor descalificați sau suspendați, precum și a altor persoane de interes, cum ar fi persoanele cu mandate de arestare emise pe numele acestora) [6],
- Colectarea taxelor de drum sau a taxelor de parcare,
- Monitorizarea traficului [7],

Figură 1 – Exemplu de sistem ANPR [1]

- Monitorizarea respectării vitezei legale în trafic [8],
- Colectarea de date ce pot fi folosite ulterior în cazuri penale,
- Servicii de securitate pentru întreprinderi [9] (exemplu: înregistrarea numerelor de înmatriculare a mașinilor ce părăsesc stațiile de alimentare fără a plăti),
- Instrumente de marketing personalizat și publicitate direcționată [10],
- Recunoașterea automată a vizitatorilor pe domenii private,
- Testarea automată a emisiilor din trafic [11],
- Calculul duratei necesare unei călătorii,
- Etc.

1.1 CONTEXT

Nevoia și importanța acestui proiect rezultă în urma statisticilor de siguranță civilă realizate pe teritoriul României.

În primul rând, aproape 6000 de copii au dispărut de acasă în anul 2021 [12]. Dintre aceștia, 19% sunt răpiți în contextul proceselor de stabilire a custodiei parentale, iar, în 20% din cazuri, apelurile din presă și sistemele de alertă pentru copii au jucat un rol important în găsirea copiilor dispăruți [13]. Alerte pot fi lansate și pe baza numărului de înmatriculare în cazul răpirii cu ajutorul unui vehicul, autoritățile putând lansa mesaje de atenționare folosind chiar platforma ROALERT. Vehiculul în cauză ar putea fi detectat de sisteme de detecție automată a plăcuței de înmatriculare.

De asemenea, în România, aproape zilnic sunt difuzate știri noi cu privire la diferite evenimente de violență fizică în trafic [14] [15] [16]. De foarte multe ori, șoferi agresivi urmăresc victima până ce ajung într-un punct în care atacul poate fi realizat cu ușurință. Un sistem de detecție a potențialilor urmăritori avertizează viitoarea victimă, iar aceasta, la rândul ei, poate lua măsurile necesare de siguranță.

Adițional, tot mai des sunt prinși la volan șoferi fără permis sau cu permisul de conducere suspendat sau chiar anulat [17] [18]. Acești șoferi ar putea fi detectați în trafic de către forțele de ordine, folosind baze de date privind posesorii unor astfel de autovehicule.

O altă statistică arată că violența domestică este un fenomen ce ia amploare în România. În ultimii 8 ani (luând ca reper anul 2021), 426 de femei au fost ucise în România

în urma violenței domestice (de partenerii lor, de foștii parteneri sau de membri ai familiei), potrivit datelor oferite de Centrul Filia, ONG care luptă pentru drepturile femeilor și pentru combaterea violenței domestice [19]. În aceeași perioadă de timp, peste 1000 de apeluri au fost preluate de operatorii din centrele împotriva violenței domestice [20]. Un alt procent îngrijorător este că, potrivit Poliției Române, aproximativ 40% din ordinele de protecție emise pentru astfel de cazuri sunt încălcate [21]. Persoanele ce sunt victime ale violenței domestice ar putea beneficia de o siguranță sporită folosind sisteme de detecție automată a urmăritorilor din trafic. Un sistem care atenționează cu privire la autovehicule ce au fost marcate în trecut ca fiind periculoase ar fi ideal pentru această categorie socială. În plus, baza de date ce are centralizate vehiculele al căror număr de înmatriculare a fost detectat, data detecțiilor și recurența acestora ar putea constitui o dovadă importantă pentru cererea în instanță a unui ordin de restricție.

Un alt raport îngrijorător provine de la Statista, din 2016, unde România ocupă locul al treilea în rândul țărilor europene cu cel mai mare număr de persoane traficate [22], cu 757 de victime raportate. Acest număr devine alarmant în comparație cu populația României (757 de victime la o populație de 19 de milioane de locuitori), raportându-ne, de exemplu, la Marea Britanie (1358 de victime la o populație de 67 de milioane de locuitori). Și aici, ca și în cazul copiilor răpiți, sistemele de detecție a numerelor de înmatriculare pot juca un rol principal în combaterea acestui tip de infracțiune.

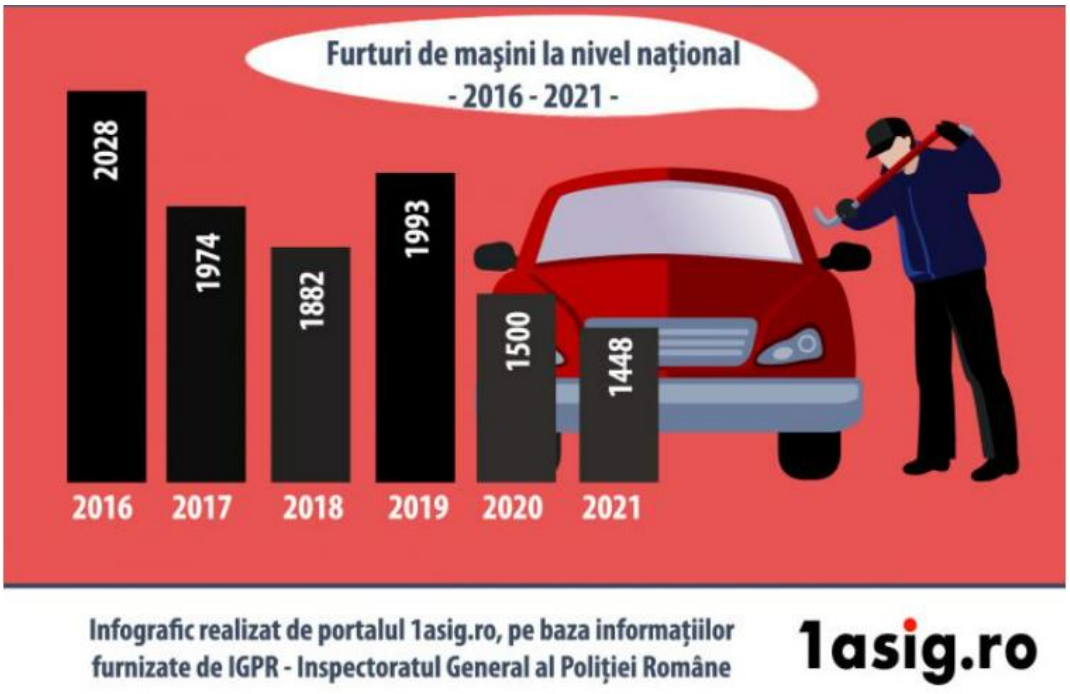
Figură 2 – Topul țărilor europene cu privire la victimele traficului de persoane [22]

Neliniștitor este și faptul că, din cele 41 de județe ale României plus zona București, în 15 dintre acestea rata criminalității este ridicată, adică în aproape o treime, conform Asociației Naționale a Evaluatorilor de Risc la Securitatea Fizică din România [23]. Detecția urmăritorilor în timp real poate oferi o siguranță în viața cotidiană a fiecărui cetățean român ce se deplasează cu un autovehicul.

Coeficienţii de criminalitate specifică la nivel de judeţe pentru anul 2020			
Nr. crt.	Județul	Coeficient de criminalitate județean (%)	Domeniu de încadrare a ratei de criminalitate
TOTAL	Media criminalității naționale 2020	502.95	
1	AB	56.67	SCĂZUT
2	AR	72.17	SCĂZUT
3	AG	79.93	MEDIU
4	BC	130.43	RIDICAT
5	BH	87.28	MEDIU
6	BN	40.16	SCĂZUT
7	BT	49.71	SCĂZUT
8	BV	143.35	RIDICAT
9	BR	74.56	SCĂZUT
10	BZ	57.86	SCĂZUT
11	CS	56.47	SCĂZUT
12	CL	23.26	SCĂZUT
13	CJ	149.32	RIDICAT
14	CT	155.88	RIDICAT
15	CV	21.47	SCĂZUT
16	DB	41.36	SCĂZUT
17	DJ	104.58	RIDICAT
18	GL	105.97	RIDICAT
19	GR	43.54	SCĂZUT
20	GJ	28.43	SCĂZUT
21	HR	22.47	SCĂZUT
22	HD	99.21	MEDIU
23	IL	45.53	SCĂZUT
24	IS	199.82	RIDICAT
25	IF	147.93	RIDICAT
26	MM	65.61	SCĂZUT
27	MH	16.70	SCĂZUT
28	MS	113.33	RIDICAT
29	NT	74.16	SCĂZUT
30	OT	59.85	SCĂZUT
31	PH	165.42	RIDICAT
32	SM	34.99	SCĂZUT
33	SJ	28.03	SCĂZUT
34	SB	106.37	RIDICAT
35	SV	78.54	MEDIU
36	TR	39.96	SCĂZUT
37	TM	139.97	RIDICAT
38	TL	25.05	SCĂZUT
39	VS	74.76	SCĂZUT
40	VL	58.65	SCĂZUT
41	VN	48.91	SCĂZUT
42	DGPMB	1032.30	RIDICAT

Tabel 1 – Rata criminalități în România, pe județe [23]

De asemenea, pe site-ul oficial al Poliției Române sunt aproximativ 22000 de mașini listate ca fiind furate [24]. Dintre acestea, 1448 au fost marcate ca fiind furate în anul 2021 [25]. Adițional, în același an, au fost identificate la intrarea în România peste 150 de mașini furate de către polițiștii de frontieră [26]. Autovehiculele furate ar putea fi detectate mai facil prin utilizarea unor software-uri ANPR, fără a fi necesară introducerea într-o bază de date a unor informații din talonul autovehiculului.



Figură 3 – Numărul de mașini furate în România în perioada 2016-2021 [25]

"We can only see a short distance ahead, but we can see plenty there that needs to be done."

"Putem vedea doar o mică distanță în viitor, dar putem vedea multe acolo care trebuie făcute."

- Alan Turing, Computing machinery and intelligence [27].

Probleme sociale existente pe teritoriul României impun existența și utilizarea unor sisteme inteligente de detecție a numerelor de înmatriculare, dar și de detecție în timp real a potențialilor urmăritori din trafic. Deoarece societatea avansează într-un ritm alert, populația are nevoie de soluții rapide, inteligente, eficiente, care să fie bazate pe tehnologii actuale, de ultimă oră. Astfel, Inteligența Artificială joacă un rol important în soluționarea diferitelor probleme umane.

„One day the AIs are going to look back on us the same way we look at fossil skeletons on the plains of Africa. An upright ape living in dust with crude language and tools, all set for extinction."

„Într-o zi, mașinăriile echipate cu Inteligență Artificială se vor uita înapoi la noi în același mod în care noi ne uităm la scheletele fosile de pe câmpiile Africii. O maimuță ce stă drept pe două picioare, care trăiește în praf, cu un limbaj grosolan și unelte grosolane, totul pregătit pentru a o conduce la dispariție."

- Nathan Bateman, Ex Machina (Movie 2014) [28].

Noi, ca specie colectivă, am visat din totdeauna să creăm mașinării inteligente, capabile de a procesa informație și de a lua decizii singure. Odată cu progresele recente din cercetare, tehnologie și odată cu îmbunătățirea puterii de calcul, inteligența artificială (în engleză, Artificial Intelligence - AI), învățarea automată (în engleză, Machine Learning - ML) și învățarea profundă (în engleză, Deep Learning – DL) au devenit un punct focal în rândul tehnologilor și al populației extinse. Deși viitorul promis al Hollywood-ului este sub semne de întrebare încă, am început să vedem și să folosim frânturi ale sistemelor inteligente în viața noastră de zi cu zi. De la asistenți virtuali inteligenți, cum ar fi Google Now, Siri, Alexa sau Cortana, până la mașini cu conducere autonomă, acceptăm treptat astfel de tehnologii inteligente în rutina noastră zilnică.

Trăim în „epoca datelor", ce deține o putere de calcul mai bună și mai multe resurse de stocare. Aceste date sau informații cresc pe zi ce trece, dar adevărata provocare este să dăm sens tuturor datelor.

Învățarea automată schimbă rapid lumea, prin diverse tipuri de aplicații și cercetări efectuate în industrie și mediul academic. Învățarea automată afectează în mod pozitiv fiecare parte a vieții noastre de zi cu zi. De la asistenți virtuali și modele de învățare automată folosite pentru a ne verifica calendarul și a reda muzică, până la reclame personalizate (care sunt atât de precise încât pot prezice de ce vom avea nevoie înainte de a ne da noi seama), învățarea automată este un factor în viețile noastre de zi cu zi, în aproape orice moment.

Detecția Automată a Numerelor de Înmatriculare este un domeniu important de cercetare în domeniul Sistemelor Inteligente de Transport (în engleze, Intelligent Transport Systems - ITS). Toate vehiculele din lume au plăcuțele de înmatriculare ca identificator principal. Cu dezvoltarea rapidă a tehnologiei de viziune artificială, metode robuste de detectare automată a obiectelor sunt introduse în ITS, iar o componentă integrală a domeniului ITS este Detecția Automată a Numerelor de Înmatriculare. [29]

Proiectul de față utilizează conceptele de Inteligență Artificială, Învățare Automată și Învățare prin Transfer pentru a crea un sistem inteligent, ce detectează nu doar numerele de înmatriculare, dar și potențialii urmăritori din trafic. Prin preluare de date în timp real, folosind un sistem Raspberry Pi, utilizatorul va primi alerte via email cu privire la eventualele pericole reprezentate de alți participanți din trafic. De asemenea, va putea vizualiza datele preluate și procesate într-o interfață grafică.

1.2 OBSTACOLE

Tehnologia ANPR trebuie să țină cont de diferențele plăcuțelor de înmatriculare de la un loc la altul pentru recunoașterea optică a caracterelor (OCR). Când plăcuțele de înmatriculare olandeze au trecut la un stil diferit în 2002, una dintre modificările făcute a fost la font, introducându-se mici spații goale în unele litere (cum ar fi P și R) pentru a le face mai distincte și, prin urmare, mai lizibile pentru astfel de sisteme. Unele standarde naționale pentru realizarea numerelor de înmatriculare folosesc variații ale dimensiunilor fontului și ale poziționării, deci sistemele ANPR trebuie să poată procesa toate aceste diferențe pentru a fi cu adevărat eficiente. Sistemele mai complicate pot face față variantelor internaționale, dar majoritatea sistemelor sunt adaptate individual pentru fiecare țară [30].

În anii 1990, progresele semnificative ale tehnologiei au condus la evoluția sistemelor ANPR de la aplicații limitate, costisitoare și greu de configurat, pe bază de aplicații fixe, la aplicații mobile simple. Aceste progrese tehnologice le-au permis polițiștilor să patruleze zilnic cu beneficiul citirii plăcuțelor de înmatriculare în timp real.

În ciuda aparentei lor eficiențe, există totuși provocări demne de notat legate de sistemele mobile. Una dintre cele mai mari provocări este că procesorul și camerele trebuie să funcționeze suficient de repede pentru a face față unor viteze de deplasare de peste 160 km/h [13a]. Echipamentul de detecție automată trebuie să fie, de asemenea, foarte eficient din punct de vedere al consumului, deoarece sursa de alimentare este, de obicei, o baterie uzuală sau bateria vehiculului. Echipamentul trebuie să fie, în mod ideal, de dimensiuni mici, pentru a fi cât mai portabil.

Algoritmul folosit trebuie, de asemenea, să poată compensa toate variabilele ce pot afecta capacitatea sistemului de detecție de a produce o citire precisă, cum ar fi:

- Momentul zilei (dimineață, prânz, amurg, seară, noapte),
- Vremea (condiții meteo favorabile și nefavorabile),
- Unghiurile dintre camera sistemului și plăcuțele de înmatriculare de procesat,
- Rezoluția slabă a fișierului, de obicei deoarece numărul de înmatriculare se află la o distantă mare față de sistem,
- Imaginile captate neclare datorită faptului că sistemul funcționează asupra unor obiecte aflate în mișcare
- Iluminarea slabă și contrastul scăzut din cauza supraexpunerii, a reflexiei sau a umbrelor,
- Obstrucționarea numărului de înmatriculare de către un obiect, cum ar fi o bară de remorcare sau praf, noroi, zăpadă,
- Schimbarea direcției vehiculului în timpul citirii plăcuței de înmatriculare, de obicei în timpul unor viraje

Deși unele dintre aceste probleme pot fi corectate în pe partea de software, în general rezolvarea obstacolelor revine părții de hardware a sistemului. Creșterea înălțimii camerei poate înlătura, de exemplu, problema obiectelor care întunecă numărul de înmatriculare, dar introduce alte probleme, cum ar fi necesitatea reajustării poziției pentru înclinarea crescută a plăcii. Cele mai multe sisteme avansate din punct de vedere tehnic au o flexibilitate crescută și sunt configurabile, fiind posibilă configurarea unui mai mare număr de camere, uzual între una și patru camere, care pot fi aranjate cum este necesar.

Conceptul de confidențialitate a datelor a cauzat îngrijorări cu privire la ANPR, cum ar fi urmărirea guvernamentală a locației cetățenilor, identificarea greșită, ratele ridicate de eroare și creșterea cheltuielilor guvernamentale. Criticii au descris-o ca pe o formă de supraveghere în masă, susținând, de asemenea, că această tehnologie este folosită pentru a crește veniturile statului, nu pentru a promova siguranța [31].

În cadrul Uniunii Europene, GDPR este legea privind protecția datelor și confidențialitatea care se aplică tuturor persoanelor rezidente. Legea GDPR fost instituită la pentru a le permite cetățenilor Uniunii Europene să-și controleze datele personale. Instituirea acestei legi impune ca orice utilizator al tehnologiei ANPR, fie entități publice sau private, să informeze persoanele că sistemul există, să pună în aplicare o evaluare a riscurilor, să dea curs cererilor de date cu caracter personal, să asiste forțele de ordine în diferite solicitări și să monitorizeze conformitatea pentru orice subcontractanți al produsului [32].

“The Best for the Group comes when everyone in the group does what's best for himself AND the group.”

„Situația cea mai bună pentru grup vine atunci când toată lumea din grup face ceea ce este mai bine pentru sine ȘI pentru grup.”

— John Nash

Realizând această lucrare și, implicit, consultând diferite opinii privind conceptul de ANPR, concluzia trasă a fost că trebuie ajuns la un compromis. Prelucrarea și stocarea datelor de către ANPR poate răni orgolii individuale și crea scenarii paranoice. În schimb, folosind această tehnologie, putem mări siguranța colectivă și individuală. Într-un scenariu ideal, unde legile sunt respectate, iar rata criminalității nu depășește pragul de nulitate, software-urile ANPR ar fi, fără loc de discuție, un abuz asupra vieții private. Însă, în realitatea cotidiană, acest tip de software-uri sporesc siguranța și obligă cetățenii la respectare legilor. Consider tehnologiile ANPR nu doar folositoare, ci necesare.

1.3 OBIECTIVE

Obiectivul principal este realizarea unui sistem inteligent de detecție a urmăritorilor în trafic, care ar putea fi transformat și folosit și în alte scopuri, cu alte opțiuni și facilități. Sporirea securității cetățeanului este prioritatea lucrării. Folosirea unui program modern, ce

ar putea fi portat și pe telefoane mobile, dar și integrat în cadrul altor aplicații deja existente, poate fi folosit de oricine, fie utilizatorul un organ al legii, sau un cetățean obișnuit.

1.4 STRUCTURA LUCRĂRII

În cadrul primului capitol, Introducere, se prezintă tema lucrării, istoricul conceptului de ANPR, contextul din care rezultă necesitatea sistemului propus, obstacolele generale și obiectivele propuse.

În continuare, în cel de al doilea capitol, Analiza și Specificarea Cerințelor Funcționale/Nefuncționale, sunt prezentate categoriile de utilizatori și cerințele de dezvoltare, împreuna cu analiza SWOT.

Al treilea capitol, Abordări Existente, se concentrează asupra produselor comerciale similare existente deja pe piață și asupra algoritmilor uzuali pentru realizarea unui astfel de sistem.

Mai departe, în al patrulea capitol, Soluția Propusă, sunt descrise tehnologiile utilizate și arhitectura sistemului, iar realizarea propriu-zisă a sistemului este descrisă în capitolul cu numărul cinci, Detalii de Implementare.

Rezultatele sunt expuse în capitolul cu numărul șase, Evaluarea Rezultatelor, unde sunt prezentate procesul de testare și de evaluare.

Posibilele dezvoltări anterioare și stadiul actual al dezvoltării reprezintă punctul de încheiere, capitolul al șaptelea, Concluzii.

2. ANALIZA ȘI SPECIFICAREA CERINȚELOR FUNCȚIONALE

Am utilizat Analiza SWOT (Strengths, Weaknesses, Opportunities, Threats – Puncte tari, Puncte slabe, Oportunități, Amenințări) pentru pasul de analiză și planificare a strategiei de dezvoltare. Cu ajutorul acestei analize (Figura 4), am determinat punctele tari, punctele slabe, oportunitățile și amenințările în legătură cu dezvoltarea sistemului. Pornind de la aceste puncte, au fost determinate cerințele pe care trebuie să le satisfacă proiectul, și anume să fie portabil, simplu de înțeles și simplu de folosit. Utilizatorul nu este obligat să introducă input-uri complicate sistemului sau să înțeleagă concepte precum cele de Învățare Automată sau Procesare de Imagine.

Figură 4 – Analiza SWOT a sistemului

2.1 DESCRIEREA CATEGORIILOR DE UTILIZATORI

Categoria amplă de utilizatori este aceea a persoanelor ce vor să își îmbunătățească securitatea personală în trafic și nu numai. Spectrul larg cuprinde categoria populației române posesoare de carnet de conducere. Publicul țintă specific este cel al persoanelor ce ar putea folosi aplicația astfel:

- Cetățeni prevăzători, care vor sa profite de orice tip de tehnologie ce le poate îmbunătăți securitatea,
- Cetățeni interesați de ultimele tehnologii în materie de gadget-uri,
- Companii ce vor să asigure securitatea mașinilor de serviciu oferite angajaților,
- Părinți ce vor să îmbunătățească siguranța copiilor în trafic, odată ce aceștia ating vârsta de 18 ani,
- Persoane ce sunt amenințate, care își doresc să fie avertizate asupra anumitor autovehicule întâlnite în trafic,
- Victime ale violenței domestice sau de alt tip, care necesită dovezi privind hărțuirea asupra căreia sunt supuse, dovezi ce pot facilita obținerea unui ordin de restricție.

2.2 CERINȚE DE SISTEM (HARDWARE ȘI SOFTWARE)

Pentru a executa codul sursă (încărcarea setului de date, antrenarea modelului, procesul de detecție și interfața grafică), sunt necesare o cameră web și un computer ce are instalată varianta Python 3.10[1] sau o variantă superioară și o listă de pachete și framework-uri (cele principale fiind Tensorflow[2], SciPy[3], Protobuf[4], GPUtil[5], AvroPython3[6], OpenCV[7], NumPy[8], Matplotlib[9], EasyOCR[10], Pytorch[11], Jupyter[12], Django[13]).

Computer-ul folosit este un Laptop Lenovo ThinkPad T14s, cu un procesor Intel Core i7-1165G7, cu 4 nuclee și o frecvență nominală de 2.8 GHz, cu o placă video integrată Intel Iris Xe, cu o capacitate a memoriei de 16 GB și un SSD cu capacitate de 512 GB, funcționând pe sistemul de operare Windows 10.

[1] Python. Disponibil: https://www.python.org/downloads/ [Accesat: 08.06.2022]
[2] Tensorflow. Disponibil: https://www.tensorflow.org/ [Accesat: 08.06.2022]
[3] SciPy. Disponibil: https://scipy.org/ [Accesat: 08.06.2022]
[4] Protobuf. Disponibil: https://pypi.org/project/protobuf/ [Accesat: 08.06.2022]
[5] GPUtil. Disponibil: https://pypi.org/project/GPUtil/ [Accesat: 08.06.2022]
[6] AvroPython3. Disponibil: https://pypi.org/project/avro-python3/ [Accesat: 08.06.2022]
[7] OpenCV. Disponibil: https://pypi.org/project/opencv-python/ [Accesat: 08.06.2022]
[8] NumPy. Disponibil: https://numpy.org/ [Accesat: 08.06.2022]
[9] Matplotlib. Disponibil: https://matplotlib.org/ [Accesat: 08.06.2022]
[10] EasyOCR. Disponibil: https://github.com/JaidedAI/EasyOCR [Accesat: 08.06.2022]
[11] Pytorch. Disponibil: https://pytorch.org/ [Accesat: 08.06.2022]
[12] Jupyter. Disponibil: https://jupyter.org/ [Accesat: 08.06.2022]
[13] Django. Disponibil: https://www.djangoproject.com/ [Accesat: 08.06.2022]

De asemenea, s-a folosit, în locul unui computer, un sistem Rasbperry Pi Model 3 B[14], cu următoarele specificații:

- SoC Broadcom BCM2837,
- procesor ARM Cortex-A53, 4 nuclee, 1,2 GHz (64/32-bit),
- 0,85 GB RAM,
- GPIO 40 Pins,
- 4 x USB 2.0,
- 4 Pole Stereo Output,
- port HDMI,
- 10/100 Ethernet,
- Micro SD Card Slot,
- BCM43143 WiFi și Bluetooth Low Energy (BLE).

2.3 CERINȚE FUNCȚIONALE/NEFUNCȚIONALE

Cerințe funcționale:

- Sistemul trebuie să avertizeze utilizatorii via email asupra posibililor urmăritori din trafic,
- Sistemul trebuie să aibă o interfață grafică minimală pentru vizualizarea detecțiilor.

Cerințe nefuncționale:

- Sistemul trebuie să poată fi utilizat fără antrenarea modelului de către utilizator,
- Sistemul trebuie să permită antrenarea modelului folosind date noi.

2.4 MODELĂRI ALE SISTEMULUI

Modelarea sistemului are ca bază procesul de schimb de informații dintre aplicație și utilizator. Utilizatorul antrenează modelul (sau folosește modelul gata antrenat), oferă sistemului datele de contact pentru emiterea avertizărilor, iar aplicația oferă ieșirea sistemului (detecțiile, avertizările, interfața grafică) folosind modelul antrenat.

a) Actorii și Cazurile de Utilizare

Actorii reprezintă utilizatorii aplicației, acei utilizatori ce își doresc a primi avertizări cu privire la posibilele autoturisme ce îi urmăresc în trafic.

Cazurile de utilizare sunt:

[14] Raspberry Pi Model 3 B. Disponibil: https://www.raspberrypi.com/products/raspberry-pi-3-model-b/ [Accesat: 23.06.2022]

- Antrenarea modelului
- Folosirea sistemului în scopul detecției urmăritorilor în trafic

b) Descrierea Cazurilor de Utilizare ale Sistemului

1. Antrenarea modelului:
   a) Precondiție: Software-ul este pornit.
   b) Flux de bază:
      i. Se încarcă seturile de date de antrenare și testare în scopul prelucrării acestora,
      ii. Se inițializează modelul,
      iii. Se antrenează modelul,
      iv. Se oferă datele de contact pentru emiterea de alerte.
   c) Cazuri alternative:
      i. Seturile de date sunt invalide -> Eroare de execuție.
      ii. Parametrii modelului sunt incorecți -> Eroare de execuție.
      iii. Model neinițializat -> Eroare de execuție.
   d) Rezultat – Modelul este antrenat, iar utilizatorul va putea primi atenționări cu privire la eventualii urmăritori din trafic.

2. Folosirea sistemului în scopul detecției urmăritorilor din trafic:
   a) Precondiție: Software-ul este pornit, iar modelul este antrenat.
   b) Flux de bază:
      i. Se oferă datele de contact pentru emiterea de alerte.
   c) Rezultat – Utilizatorul primește atenționări cu privire la eventualii urmăritori din trafic și poate urmări detecțiile în interfața grafică.

3. ABORDĂRI EXISTENTE

3.1 PRODUSE COMERCIALE

Dacă pentru numerele de înmatriculare de pe teritoriul american există o multitudine de produse comerciale, prezentate în continuare, pentru numerele de înmatriculare române proiecte comerciale de acest tip nu sunt cunoscute.

Plate Recognizer[15] este un produs software ce funcționează în orice mediu, optimizat pentru locația utilizatorului. Acest produs comercial returnează utilizatorului marca, modelul, culoarea, regiunea și direcția de deplasare a vehiculelor detectate. Detecția numărului de înmatriculare se poate face pe baza unei imagini sau folosind un flux de date video (chiar și în timp real). Plate Recognizer este destinat monitorizării parcărilor, autostrăzilor si colectării de taxe de drum. Serviciul de detecție în timp real, cu preluare de marcă, model, culoare, orientare și direcție de deplasare, plus accesul la interfața grafică ajunge la prețul de 60 de dolari americani pe lună, pentru un utilizator [32].

OPENALPR[16] propune un sistem asemănător, al cărui preț ajunge la 1499 de dolari americani (sistemul video plus 1 an de utilizare a software-ului dedicat) [32].

Compania **Axis**[17] propune sistemul AXIS P1455-LE-3 Plate Verifier Kit, pe care ei îl descriu ca fiind ieftin, deși funcționează perfect doar în condiții de viteză relativ mică. Sistemul, de tip fix, detectează numărul de înmatriculare și trimite o comandă pe baza acestuia (exemplu: deschidere poartă sau alertă mașină necunoscută). Produsul, pe site-ul devodep.ro, are prețul de 5692 RON [33].

Compania **Leonardo**[18] a dezvoltat soluția mobilă ELSAG Mobile Plate Hunter, care declanșează alerte cu privire la vehicule marcate anterior din baza de date. Acest produs funcționează, însă, perfect doar pentru vehiculele înmatriculate în Statele Unite ale Americii și pentru alte câteva țări [34].

Prețul oficial nu este afișat pe site-ul acestora, însă variante la mâna a doua se comercializează pe ebay.com la prețul de aproximativ 270 de dolari americani [35].

[15] Plate Recognizer. Disponibil: https://platerecognizer.com/ [Accesat: 08.06.2022]
[16] OpenALPR. Disponibil: https://www.openalpr.com/ [Accesat: 08.06.2022]
[17] Axis. Disponibil: https://www.axis.com/ [Accesat: 08.06.2022]
[18] Leonardo. Disponibil: https://leonardocompany-us.com/ [Accesat: 08.06.2022]

Piața de sisteme pentru detecția numerelor de înmatriculare este nu doar performantă, dar și competitivă în spațiul american. Numitorul comun al tuturor aceste sisteme comerciale existente este prețul ridicat, inaccesibil pentru utilizatorul român. Astfel, piața de servicii software și produse fizice pentru detecția numerelor de înmatriculare române prezintă un potențial pentru devoltatorii din industria IT și Automotive.

3.2 METODE EXISTENTE

Există șapte algoritmi principali ce sunt folosiți, în general, pentru realizarea software-ului de identificare a unei plăcuțe de înmatriculare:

- Algoritmul de localizare a placuței de înmatriculare – responsabil pentru găsirea și izolarea numărului de înmatriculare de pe imagine,
- Algoritmul de corecție pentru orientarea și dimensionarea plăcuței de înmatriculare - compensează înclinarea numărului de înmatriculare și ajustează dimensiunile la dimensiunea necesară,
- Algoritmul de normalizare – se ocupă cu ajustarea luminozității și a contrastului imaginii captate,
- Algoritmul de segmentare optică a caracterelor – are rolul de identifica fiecare caracter alfanumeric individual de pe numărul de înmatriculare,

Figură 5 – Exemplu de segmentare a caracterelor [1]

- Algoritmul de recunoaștere optică a caracterelor – are scopul de a traduce fiecare caracter detectat pe plăcuța de înmatriculare,
- Algoritmul de analiză sintactică/geometrică – se ocupă de verificarea caracterelor și a pozițiilor acestora în raport cu regulile specifice fiecărei țări,
- Algoritmul de mediere a valorii recunoscute pe mai multe imagini – cu scopul de a produce un rezultat cu o acuratețe ridicată, luând în considerare faptul că fiecare imagine capturată poate conține o lumină reflectată, poate fi parțial umbrită sau poate fi obstrucționată din diverse motive [13a].

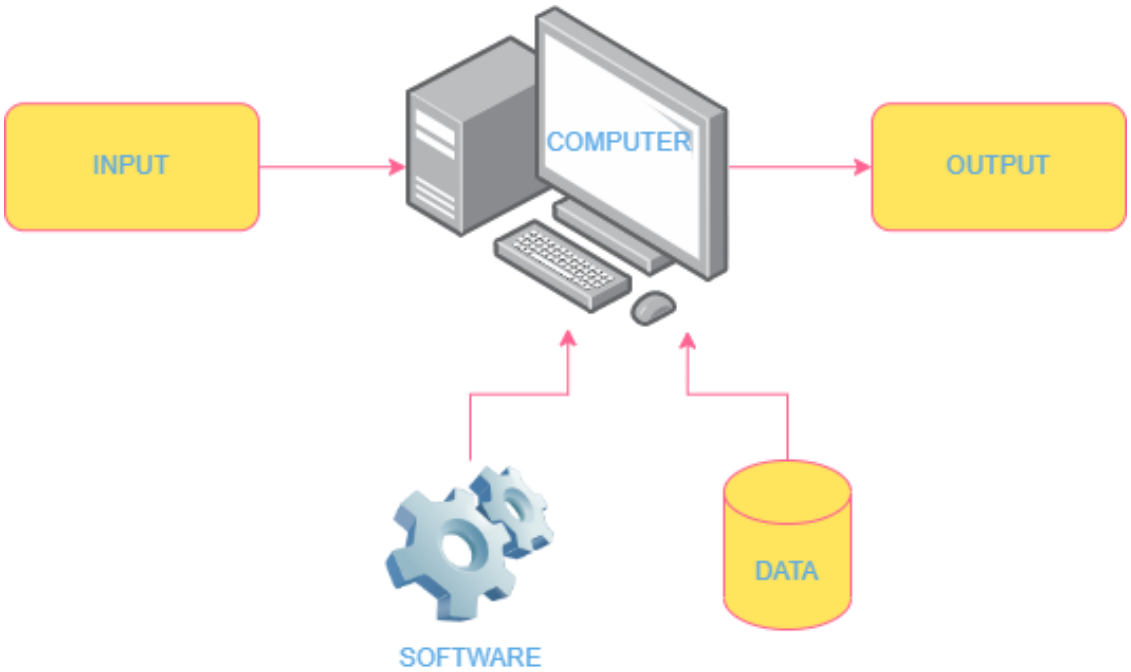
4. SOLUȚIA PROPUSĂ

4.1 Tehnologii utilizate

Lucrarea de față folosește Python3[19], Tensorflow[20], Învățarea Automată și o multitudine de pachete, biblioteci și Framework-uri Python pentru a implementa soluția propusă.

Sistemul de operare utilizat este Windows 10[21]. Am folosit, de asemenea, acolo unde era cazul, posibilitatea de a rula instrucțiuni specifice sistemului de operare Linux, pentru a îmbunătăți portabilitatea aplicației.

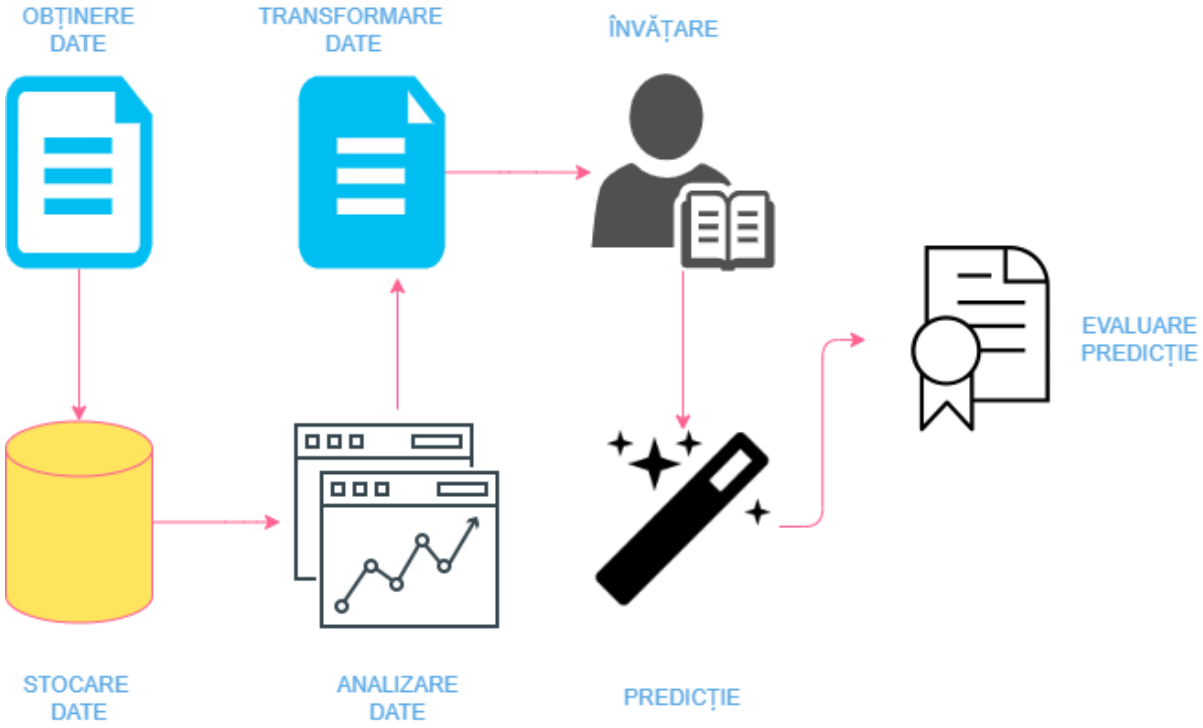
Învățarea automată (în engleză, Machine Learning – ML) este un vast domeniu al informaticii ce utilizează tehnici statistice pentru a le oferi programelor abilitatea de a învăța din experiențele trecute și de a îmbunătăți modul în care îndeplinesc anumite sarcini, fără a fi programate în mod explicit și fără intervenție umană.



Figură 6 – Procesul de Învățare Automată

[19] Python. Disponibil: https://www.python.org/ [Accesat: 08.06.2022]
[20] Tensorflow. Disponibil: https://www.tensorflow.org/ [Accesat: 08.06.2022]
[21] Windows 10. Disponibil: https://www.microsoft.com/ro-ro/windows/windows-10-specifications [Accesat: 08.06.2022]

Practic, sistemele informatice pot oferi un sens datelor în același mod în care o ființă umană ar face același lucru. Astfel, programele de tip ML construiesc modele matematice din date brute folosind algoritmi și metode, iar, pe baza acestor modele, realizează predicții.



Figură 7 – Procesul de învățare Automată (detalii)

"Can machines think?"
„Pot mașinăriile să gândească?”

- Alan Turing, "Computer Machinery and Intelligence,", 1950 [27]

“A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.”

„Se spune că un program de calculator învață din experiența E cu privire la o anumită clasă de sarcini T și cu măsura performanței P, dacă performanța sa la sarcinile din T, măsurată de P, se îmbunătățește cu experiența E.”

- Tom M. Mitchell [36]

Transfer Learning: Modelelor de rețele neuronale convoluționale profunde le pot fi necesare zile sau chiar săptămâni pentru a se antrena pe seturi de date foarte mari. O modalitate de a scurta acest proces este de a reutiliza modele pre-antrenate, care au fost dezvoltate pentru seturi de date standard de Computer Vision. Modele performante (uzual, antrenate pe seturi de date mari) pot fi descărcate și utilizate direct sau integrate într-un model. Astfel, învățarea prin transfer implică utilizarea modelelor instruite pe o problemă ca

punct de plecare pentru o altă problemă conexă. Conceptul de Transfer Learning este foarte util în momentul în care problema de tratat are un set de date minimal.

Învățarea prin transfer este flexibilă. Modelul pre-antrenat poate fi folosit ca un program separat de extragere a caracteristicilor, caz în care input-ul poate fi preprocesat de model sau de către o porțiune a modelului la un output dat pentru fiecare imagine de intrare, care poate apoi fi utilizată ca intrare atunci când este antrenat un model nou [37].

Alternativ, modelul pre-antrenat sau porțiunea dorită a modelului poate fi integrată direct într-un nou model de rețea neuronală. În această utilizare, ponderile pre-antrenate pot fi înghețate, astfel încât să nu fie actualizate pe măsură ce noul model este antrenat. Alternativ, ponderile pot fi actualizate în timpul antrenării noului model, dar eventual cu o rată de învățare mai mică, permițând modelului pre-antrenat să acționeze ca o schemă de inițializare a ponderilor atunci când antrenează noul model [37].

Python[22] este un limbaj de programare interpretat, cu sursă deschisă, gratuit, popular, care are capabilitățile unui limbaj de programare de nivel înalt. Sintaxa ușor de învățat și capacitatea de portabilitate îi oferă o mare popularitate în zilele noastre.

Python are o mulțime de puncte forte, printre care și:

- Este ușor de învățat și de înțeles, dispunând de o sintaxă simplă.
- Este un limbajul multifuncțional, deoarece acceptă programarea structurală, programarea orientată pe obiecte, precum și programarea funcțională.
- Dispune de un număr mare de module ușor disponibile pentru utilizare, transformând Python într-un limbaj extensibil.
- Dispune de un suport mare al pentru comunității - Fiind un limbaj de programare open source, Python este susținut de o comunitate foarte mare de dezvoltatori.
- Este un limbaj scalabil, deoarece oferă o structură îmbunătățită pentru programele mai mari decât scripturile shell.

Cu toate acestea, deși Python este un limbaj de programare popular și puternic, are propria sa slăbiciune, aceasta fiind reprezentată de viteză lentă de execuție. Această viteză lentă de execuție rezultă din faptul că Python este un limbaj interpretat, și nu unul compilat.

Python ne oferă posibilitatea de a crea un program de detecție și recunoaștere a numerelor de înmatriculare, deoarece are un set extensiv de pachete gata de folosit, multe reprezentând un mare ajutor în domeniul ML, precum Numpy[23], Seaborn[24], Matplotlib[25], Pandas[26] si Scikit-learn[27]. De asemenea, Python este potrivit aplicațiilor de tip ML, deoarece, cu ajutorul acestuia, putem manipula și analiza date, extrage parametrii datelor,

[22] Python. Disponibil: https://www.python.org/ [Accesat: 08.06.2022]
[23] Numpy. Disponibil: https://numpy.org/ [Accesat: 09.06.2022]
[24] Seaborn. Disponibil: https://seaborn.pydata.org/ [Accesat: 09.06.2022]
[25] Matplotlib. Disponibil: https://matplotlib.org/ [Accesat: 09.06.2022]
[26] Pandas. Disponibil: https://pandas.pydata.org/ [Accesat: 09.06.2022]
[27] Scikit-learn. Disponibil: https://scikit-learn.org/stable/ [Accesat: 09.06.2022]

manipula, evalua și îmbunătăți datele. Așadar, realizăm aplicația folosind Python și mai multe dintre bibliotecile sale.

Mediul de dezvoltare folosit a fost Pycharm IDE[28], deoarece este disponibil gratuit, are facilități suplimentare la conectarea cu adresa de email academică, are integrat sistemul de versionare Git și este potrivit pentru utilizatorii Windows sau Linux.

De asemenea, am folosit Jupyter[29]. Notebook-urile Jupyter oferă un mediu computațional interactiv web pentru dezvoltarea aplicațiilor Data Science bazate pe Python. Jupyter Notebook-urile ilustrează procesul de analiză pas cu pas prin aranjarea elementelor precum codul, imaginile, textul, rezultatele etc.

Jupyter Notebook se poate conecta la multe nuclee pentru a permite programarea în diferite limbi. Un nucleu Jupyter este un program responsabil pentru gestionarea diferitelor tipuri de solicitări (precum execuția codului) și furnizarea unui răspuns. Spre deosebire de multe alte interfețe asemănătoare notebook-urilor, în Jupyter, nucleele nu știu că sunt atașate la un anumit document și pot fi conectate la mai mulți clienți simultan. În mod implicit, Jupyter Notebook este livrat cu nucleul IPython.

Pentru a izola diversele pachete necesare, am creat un environment virtual. Acest environment face mult mai facilă gestionarea dependințelor proiectului. Virtual environment-ul a fost asociat unui ipykernel folosit pentru rularea celulelor de cod din cadrul Jupyter Notebook-ului.

OpenCV[30] (Open Source Computer Vision Library) este o bibliotecă de învățare automată cu sursă deschisă, multiplatformă, ce oferă o infrastructură de bază pentru domeniul Computer Vision. Cu ajutorul OpenCV, pot fi realizate aplicații de conducere autonomă, adnotare a imaginilor, monitorizarea în timp real a recoltelor pe bază de drone, etc.

OpenCv se concentrează, în principal, pe captarea de imagini și fișiere video, pentru a analiza caracteristici importante din cadrul acestora, cum ar fi detecția de obiecte, detecția de fețe umane, detecția de emoții, etc. De asemenea, OpenCV este un modul ce facilitează procesarea rapidă, deoarece codul său sursă este bazat pe C/C++ [38].

Astfel, OpenCv joacă un rol principal în aplicațiile de inteligență artificială bazate pe procesarea imaginilor.

Recunoașterea automată textului (cunoscută și ca recunoașterea optică a caracterelor) este procesul de recunoaștere a textului dintr-un anume scenariu prin înțelegerea și analiza modelelor sale subiacente [39].

[28] Pycharm. Disponibil: https://www.jetbrains.com/pycharm/ [Accesat: 08.06.2022]
[29] Jupyter. Disponibil: https://jupyter.org/ [Accesat: 08.06.2022]
[30] OpenCV. Disponibil: https://pypi.org/project/opencv-python/ [Accesat: 08.06.2022]

Figură 8 – Procesul de Recunoaștere Automată a Caracterelor

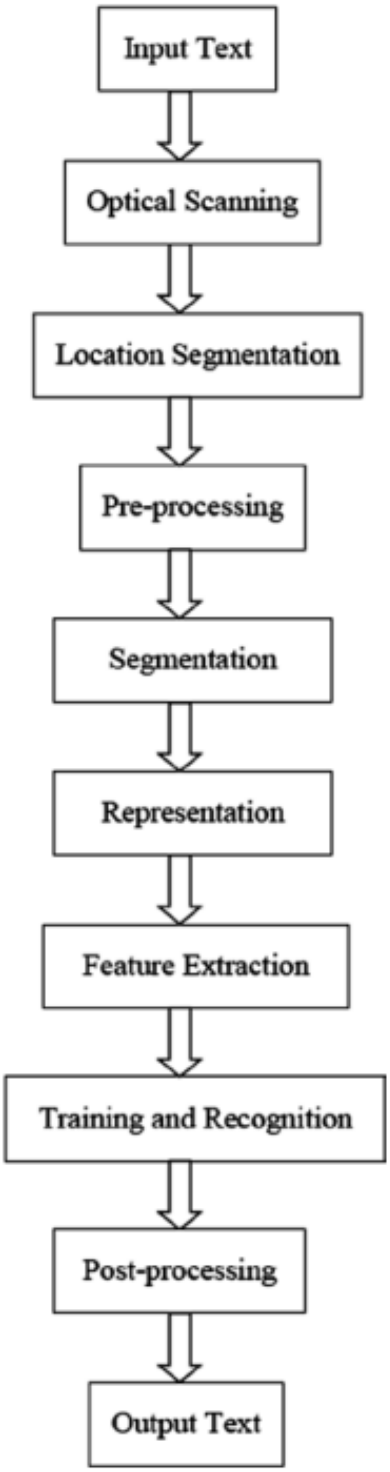
Cu alte cuvinte, sistemele OCR transformă o imagine bidimensională, care ar putea conține text tipărit sau scris de mână, din reprezentarea de tip imagine într-un text care poate fi citit de către o mașinărie. Din acest motiv, OCR este, de asemenea, recunoscut ca un subdomeniu al procesării de imagine. [40]

OCR, ca proces, constă, în general, din mai multe subprocese ce trebuie a fi realizate cât mai precis posibil.

Subprocesele acestea sunt, uzual:

- încărcarea imaginii ca input,
- scanarea imaginii (convertirea elementelor din imagine într-o matrice bidimensională de puncte albe și negre),
- localizarea zonelor de interes (zonele textului),
- preprocesarea imaginii (eliminarea zgomotului, umplerea golurilor, normalizarea și compresia imaginii),
- segmentarea caracterelor,
- recunoașterea caracterelor,
- postprocesarea rezultatului [41].

Pentru realizarea unui model de învățare automată de recunoaștere a caracterelor, sunt inserați și pașii de reprezentare, extragere de caracteristici, antrenare și testare. [40]



Figură 9 – Subprocesele pentru realizarea unui model de Recunoaștere optică a caracterelor [40]

Recunoașterea optică a caracterelor poate fi folosită în diverse aplicații, cum ar fi citirea documentelor, preluarea informațiilor scrise de mână, identificarea produselor de pe rafturi, audierea textului pentru persoanele cu dificultăți vizuale, detectarea semnelor de circulație și a mesajelor acestora pentru conducerea autonomă și multe altele [39].

Deși este folosit în multiple domenii și aplicații, OCR prezintă, încă, unele dificultăți, ce împiedică performanța. Sursa (documentul, imaginea), în cazul în care este degradată, aduce obstacole în procesarea OCR. Unele dintre caracteristicile de calitate joasă comune sunt iluminarea scăzută, contrastul insuficient între prim-plan și fundal, rezoluția scăzută, caracterele cu aspect șters, zgomotul suprapus peste imaginea captată, etc.

EasyOCR[31] este o librărie OCR optimizată pentru Python pentru extragerea textului din fișiere imagine.

SciPy[32] este un pachet Python gratuit, cu sursă deschisă, folosit pentru calcule computaționale științifice și tehnice. Conține module pentru optimizare, pentru algebră liniară, pentru integrare, interpolare, transformări, procesare de imagine și de semnal și multe alte sarcini inginerești.

PIL/Pillow[33] (Python Imaging Library) este o librărie gratuită pentru limbajul de programare Python, ce oferă suport pentru manipularea a multiple formate de imagine. Biblioteca Pillow conține funcționalități de bază de procesare a imaginilor, inclusiv operațiuni de filtrare cu un set de nuclee convoluționale încorporate și conversii în spații de culoare.

NumPy[34] (Numerical Python) este o librărie specifică limbajului Python, ce oferă suport pentru operații computaționale asupra unor matrice și vectori multidimensionali. Operații importante din cadrul pachetului NumPy sunt transformările Fourier și operații de algebră liniară. O imagine, în esență, este o matrice bidimensională sau tridimensională ce conține pixeli. Prin urmare, folosind operații de bază incluse în pachetul NumPy, precum feliere, mascare și indexare, se pot modifica valorile pixelilor unei imagini. Imaginea poate fi încărcată folosind modulul Skimage și ulterior afișată folosind modulul Matplotlib.

Matplotlib[35] este librăria specifică Python și NumPy pentru realizarea de grafice bidimensionale și tridimensionale.

Pandas[36] este pachetul Python ce face Python să fie unul dintre limbajele de programare favorite pentru Data Science. Pandas este folosit pentru încărcarea, pregătirea, manipularea, modelarea și analiza datelor. Reprezentarea datelor în Pandas se face cu ajutorul a trei structuri de date (serii, data frame-uri și panel-uri). Am folosit data frame-uri,

[31] EasyOCR. Disponibil: https://github.com/JaidedAI/EasyOCR [Accesat: 08.06.2022]
[32] SciPy. Disponibil: https://scipy.org/ [Accesat: 08.06.2022]
[33] Pillow. Disponibil: https://pypi.org/project/Pillow/ [Accesat: 08.06.2022]
[34] NumPy. Disponibil: https://numpy.org/ [Accesat: 08.06.2022]
[35] Matplotlib. Disponibil: https://matplotlib.org/ [Accesat: 08.06.2022]
[36] Pandas. Disponibil: https://pandas.pydata.org/ [Accesat: 08.06.2022]

ce reprezintă o structură de date bidimensională, ce conține date eterogene. Data frame-urile sunt folosite, în general, pentru a reprezenta date tabulare (tabele).

PyTorch[37] este o bibliotecă populară de învățare automată Python, bazată pe Torch și dezvoltată de Facebook. Torch este o bibliotecă open-source de învățare automată implementată în limbajul de programare C [42].

PyTorch deține o selecție vastă de instrumente și biblioteci ce extind domeniile de Computer Vision, procesarea limbajului natural (în engleză, Natural Language Processing - NLP) și o serie de alte programe de învățare automată [42].

Tensorflow[38] este biblioteca de bază open source pentru dezvoltarea și și antrenarea de modele ML, creată de Google. TensorFlow este o platformă open source end-to-end pentru învățarea automată. Are un ecosistem cuprinzător și flexibil de instrumente, biblioteci și resurse ale comunității, care permit cercetătorilor să promoveze tehnici de ultimă generație în ML, iar dezvoltatorilor să construiască și să implementeze cu ușurință aplicații bazate pe ML [43].

De asemenea, dispune de o arhitectură flexibilă, ceea ce înseamnă că poate fi implementat pe o mare varietate de platforme, de la procesoare și GPU-uri, până la servere de pe dispozitive mobile [44].

[37] Pytorch. Disponibil: https://pytorch.org/ [Accesat: 08.06.2022]
[38] Tensorflow. Disponibil: https://www.tensorflow.org/ [Accesat: 08.06.2022]

5. DETALII DE IMPLEMENTARE

Soluția a fost implementată într-un Jupyter Notebook, cu apel la script-uri Python și folosind framework-ul Django[39] pentru interfața grafică. Codul a fost scris în Python 3.10, folosindu-se pentru calcule și operații matematice biblioteca NumPy (pe lângă operațile puse la dispoziție de TensorFlow), iar pentru diferite reprezentări grafice folosindu-se biblioteca Matplotlib.

Modelul folosit este modelul SSD MobileNet V2 FPNLite 320x320[40], preantrenat pe setul de date COCO 2017[41].

Modelul este un model de detecție de obiecte cu următorii parametrii:

- Cu o abordare SSD (Single Shot MultiBox Detector), ce mărește viteza de procesare eliminând din rețea pasul de propunere de regiune,
- Potrivită pentru aplicații mobile,
- Cu o caracteristică de extragere a datelor FPN (Feature Pyramid Network) – Caracteristică de Rețea Piramidală), ce oferă un flux de execuție de sus în jos pentru a construi straturi de rezoluție mai mare dintr-un strat bogat semantic,
- Cu facilitate de predicție pe bază de adnotări bounding box (casetă de încadrare).

Revenind la caracteristica de extragere a detelor, FPN este un model de tip Analiza la scară multiplă, o idee destul de veche în procesarea imaginilor, ce a fost implementată în diferite arhitecturi de rețele neuronale.

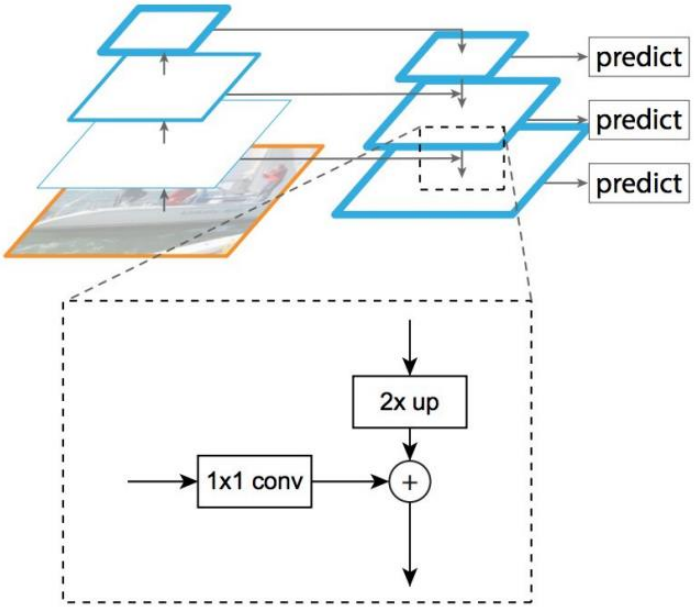
Unul dintre cele mai proeminente modele de acest fel a fost propus de Lin și colab. în [45], care au dezvoltat modelul în principal pentru detecția de obiecte, însă modelul a ajuns să prezinte o mare importanță în domeniul segmentării de imagine [45].

O arhitectură multistrat piramidală de Rețele Neuronale Convoluționale profunde a fost folosită pentru a construi această rețea piramidală. Pentru a întruni caracteristici de rezoluții scăzute și ridicate, rețeaua piramidală este compusă dintr-o cale de jos în sus, o cale de sus în jos și conexiuni laterale. Caracteristicile concatenate sunt, mai apoi, procesate de un strat convoluțional (filtru convoluțional, îmbinat cu matricea de intrare, folosite împreună pentru antrenarea ponderilor) de mărime 3 pe 3 pentru a produce rezultatul fiecărei etape. În cele din urmă, fiecare etapă parcursă a căii de sus în jos generează o predicție pentru detecția unui obiect [46].

[39] Django. Disponibil: https://www.djangoproject.com/ [Accesat: 08.06.2022]

[40] SSD MobileNet V2 FPNLite 320x320 Disponibil: http://download.tensorflow.org/models/object_detection/tf2/20200711/ssd_mobilenet_v2_fpnlite_320x320_coco17_tpu-8.tar.gz [Accesat: 08.06.2022]

[41] COCO. Disponibil: https://cocodataset.org/#home [Accesat: 08.06.2022]



Figură 10 – Conexiunea laterală și calea de sus în jos, îmbinate prin însumare [45]

Modelul ales pentru această lucrare realizează următorii pași principali:

- Extragerea caracteristicilor (procesul de transformare a datelor brute în caracteristici numerice, care pot fi procesate, păstrându-se în același timp informațiile din setul de date original),
- Codificarea casetelor de încadrare (modelul de învățare automată primește o imagine ca intrare și, pe baza acesteia, generează o propoziție care descrie o imaginea și casetele de încadrare prezente în aceasta),
- Potrivirea (procesul prin care se compară diferite valori de date în format structurat sau nestructurat, pe baza similitudinii sau a unei entități subiacente),
- Calculul similitudinii (se utilizează abordarea celui mai apropiat vecin pentru a identifica asemănarea a două sau mai multe obiecte unul cu celălalt pe baza funcțiilor algoritmice de distanță),
- Generarea variabilelor de casete de predicție (variabilele care sunt mapate la variabila țintă printr-o relație empirică determinată prin date),
- Generarea ancorelor (ancorele utilizează tehnici de învățare prin consolidare în combinație cu un algoritm de căutare bazat pe grafuri pentru a reduce numărul de apelări ale modelului și, prin urmare, timpul de rulare necesar, la minimum),
- Postprocesarea (îmbunătățirea calității imaginilor produse de către decodorul unui sistem de compresie a imaginilor cu pierderi),
- Calculul pierderii.

Preantrenarea modelului este realizată pe un număr de 50000 de pași, cu 128 de imagini procesate pe iterație, folosind un optimizator de tip momentum, ce modifică atributele rețelei neuronale (ponderile, rata de învățare) cu scopul de a minimiza pierderile și de a îmbunătăți precizia.

Setul de date folosit pentru antrenarea și testarea modelului se numește Car License Plate Detection[42], ce conține 433 de imagini cu vehicule și adnotări în formatul PASCAL VOC.

În continuare, în prezentarea codului sursă, vor fi adpugate doar frânturi de cod, pentru a nu supraîncărca această lucrare. Codul integral va fi disponibil la finalul documentației, sub formă de link către un repository GitHub.

Înainte de realizarea propriu-zisă a codului, am început cu procesul de setup, în care am creat directorul necesar, un virtual environment, am facut upgrade la utilitarul PIP și am instalat pachetul ipykernel.

Am asociat un fișier de tip Jupyer Notebook cu environment-ul creat, am lansat în execuție Jupyter și am setat kernel-ul folosit ca fiind virtual environment-ul creat. Notebook-ul Jupyter de bază a fost pus la dispoziție de către Nicholas Renotte, Data Scientist din cadrul companiei IBM[43].

În finalul incipitului, am stabilit anumite constante pentru a determina existența unui cod cât mai clar, concis și curat, cum ar fi numele modelului și link-uri către diferite resurse.

```
# File name, links that will be used frequently
CUSTOM_MODEL_NAME = 'my_ssd_mobnet'
PRETRAINED_MODEL_NAME =
'ssd_mobilenet_v2_fpnlite_320x320_coco17_tpu-8'
PRETRAINED_MODEL_URL =
'https://download.tensorflow.org/models/object_detection/tf2/20200
711/ssd_mobilenet_v2_fpnlite_320x320_coco17_tpu-8.tar.gz'
TF_RECORD_SCRIPT_NAME = 'generate_tfrecord.py'
LABEL_MAP_NAME = 'label_map.pbtxt'
```

De asemenea, am descărcat un Dataset gratuit de pe site-ul kaggle.com, Dataset numit Car License Plate Detection.

Acest Dataset l-am împărțit în două categorii, antrenare și testare (411 imagini pentru antrenare și 22 pentru testare).

[42] Car License Plate Detection . Disponibil: https://www.kaggle.com/datasets/andrewmvd/car-plate-detection?resource=download [Accesat: 08.06.2022]

[43] Real Time Automatic Number Plate Recognition, Nicholas Renotte. Disponibil: https://github.com/nicknochnack/RealTimeAutomaticNumberPlateRecognition [Accesat: 21.06.2022]

În pasul 1 al proiectului, am clonat un repository existent, deoarece dorim folosirea unui model existent preantrenat, modelul SSD MobileNet V2 FPNLite 320x320, pe care îl adaptăm cazului nostru. Această abordare este cunoscută sub numele de Transfer Learning.

Urmează instalarea modulului Tensorflow Object Detection și rularea unui script de verificare pentru a ne asigura că avem minimul de pachete necesare pentru rularea codului în continuare. Instalarea TFOD a fost adaptată și pentru utilizatorii de Linux.

```
# Installing Tensorflow Object Detection
if os.name=='posix':
    !apt-get install protobuf-compiler
    !cd Tensorflow/models/research && protoc
object_detection/protos/*.proto --python_out=. && cp
object_detection/packages/tf2/setup.py . && python -m pip install
.

if os.name=='nt':

url="https://github.com/protocolbuffers/protobuf/releases/download
/v3.15.6/protoc-3.15.6-win64.zip"
    wget.download(url)
    !move protoc-3.15.6-win64.zip {paths['PROTOC_PATH']}
    !cd {paths['PROTOC_PATH']} && tar -xf protoc-3.15.6-win64.zip
    os.environ['PATH'] += os.pathsep +
os.path.abspath(os.path.join(paths['PROTOC_PATH'], 'bin'))
    !cd Tensorflow/models/research && protoc
object_detection/protos/*.proto --python_out=. && copy
object_detection\\packages\\tf2\\setup.py setup.py && python
setup.py build && python setup.py install
    !cd Tensorflow/models/research/slim && pip install -e .
```

Rularea script-ului de verificare pus la dispoziție de Tensorflow:

```
# Verification script from Tensorflow
VERIFICATION_SCRIPT = os.path.join(paths['APIMODEL_PATH'],
'research', 'object_detection', 'builders',
'model_builder_tf2_test.py')
# Verifying installation for TFOD
!python {VERIFICATION_SCRIPT}
```

După descărcarea modelului preantrenat, în pasul 2 creăm un Label Map. În învățarea automată, etichetarea datelor (în Engleza, Data Labeling) este procesul de identificare a datelor brute (imagini, fișiere text, videoclipuri etc.) și de adăugare a uneia sau mai multor etichete semnificative și informative pentru a oferi context, astfel încât un model de învățare automată să poată învăța din acestea. În cazul de față, avem un singur label (o singură etichetă), și anume numărul de înmatriculare.

```
# Creating the Label Map
labels = [{'name':'licence', 'id':1}]

with open(files['LABELMAP'], 'w') as f:
    for label in labels:
        f.write('item { \n')
        f.write('\tname:\'{}\'\n'.format(label['name']))
        f.write('\tid:{}\n'.format(label['id']))
        f.write('}\n')
```

În pasul 3, ne ocupăm de crearea unor fișiere de tip Tensorflow records, necesare pentru configurarea și reantrenarea modelului în cadrul procesului de învățare prin transfer. Crearea acestor Tensorflow records se realizează în mod automatizat, folosind un script realizat de autorul codului schiță, Nicholas Renotte.

```
# Cloning script to generate TF records
if not os.path.exists(files['TF_RECORD_SCRIPT']):
    !git clone https://github.com/nicknochnack/GenerateTFRecord
{paths['SCRIPTS_PATH']}

# Creating TF records
!python {files['TF_RECORD_SCRIPT']} -x
{os.path.join(paths['IMAGE_PATH'], 'train')} -l
{files['LABELMAP']} -o {os.path.join(paths['ANNOTATION_PATH'],
'train.record')}
!python {files['TF_RECORD_SCRIPT']} -x
{os.path.join(paths['IMAGE_PATH'], 'test')} -l {files['LABELMAP']}
-o {os.path.join(paths['ANNOTATION_PATH'], 'test.record')}
```

Deoarece modelul descărcat este sub forma unui config file, în pasul 4 ne ocupăm de copierea acestui fișier de configurare, iar, în pasul 5, îl reconfigurăm pentru cazul nostru de utilizare.

```
# Copying model config file to training folder
if os.name =='posix':
    !cp {os.path.join(paths['PRETRAINED_MODEL_PATH'],
PRETRAINED_MODEL_NAME, 'pipeline.config')}
{os.path.join(paths['CHECKPOINT_PATH'])}
if os.name == 'nt':
    !copy {os.path.join(paths['PRETRAINED_MODEL_PATH'],
PRETRAINED_MODEL_NAME, 'pipeline.config')}
{os.path.join(paths['CHECKPOINT_PATH'])}

# Getting configs from pipeline config file
config =
config_util.get_configs_from_pipeline_file(files['PIPELINE_CONFIG'
])
```

```
# Read contents of pipeline config file
pipeline_config = pipeline_pb2.TrainEvalPipelineConfig()
with tf.io.gfile.GFile(files['PIPELINE_CONFIG'], "r") as f:
    proto_str = f.read()
    text_format.Merge(proto_str, pipeline_config)

# Update config file for transfer learning
pipeline_config.model.ssd.num_classes = len(labels)
pipeline_config.train_config.batch_size = 4
pipeline_config.train_config.fine_tune_checkpoint =
os.path.join(paths['PRETRAINED_MODEL_PATH'],
PRETRAINED_MODEL_NAME, 'checkpoint', 'ckpt-0')
pipeline_config.train_config.fine_tune_checkpoint_type =
"detection"
pipeline_config.train_input_reader.label_map_path=
files['LABELMAP']
pipeline_config.train_input_reader.tf_record_input_reader.input_pa
th[:] = [os.path.join(paths['ANNOTATION_PATH'], 'train.record')]
pipeline_config.eval_input_reader[0].label_map_path =
files['LABELMAP']
pipeline_config.eval_input_reader[0].tf_record_input_reader.input_
path[:] = [os.path.join(paths['ANNOTATION_PATH'], 'test.record')]

# Write updated configs to pipeline config file
config_text = text_format.MessageToString(pipeline_config)
with tf.io.gfile.GFile(files['PIPELINE_CONFIG'], "wb") as f:
    f.write(config_text)
```

În pasul 6, generăm folosind un script pus la dispoziție de Tensorflow comanda de antrenare a modelului, pe care o vom utiliza într-o fereastră de cmd (Command Window), pentru a vizualiza mai bine progresul procesului de antrenare.

```
# Training script from TFOD
TRAINING_SCRIPT = os.path.join(paths['APIMODEL_PATH'], 'research',
'object_detection', 'model_main_tf2.py')

# Generating training command
command = "python {} --model_dir={} --pipeline_config_path={} --
num_train_steps=10000".format(TRAINING_SCRIPT,
paths['CHECKPOINT_PATH'],files['PIPELINE_CONFIG'])
```

Am antrenat modelul folosind un număr de 10000 de pași. La fiecare 1000 de pași realizați, se creează un checkpoint. Vom folosi ultimul checkpoint creat pentru a detecta numere de înmatriculare folosind imagini (pasul 8), dar și în timp real (pasul 9), cu flux de date de la o cameră video.

Evaluarea modelului se realizează similar, folosind același script Tensorflow.

```
# Generating model evaluation command
command = "python {} --model_dir={} --pipeline_config_path={} --
checkpoint_dir={}".format(TRAINING_SCRIPT,
paths['CHECKPOINT_PATH'],files['PIPELINE_CONFIG'],
paths['CHECKPOINT_PATH'])
```

Urmează încărcarea modelului antrenat și construirea unui model de detecție, încercând de asemenea și evitarea epuizării resurselor GPU-ului, în cazul utilizatorilor cu placă video dedicată.

```
# Preventing GPU from complete consumption of resources
gpus = tf.config.list_physical_devices('GPU')
if gpus:
    try:
        tf.config.experimental.set_virtual_device_configuration(
            gpus[0],
[tf.config.experimental.VirtualDeviceConfiguration(memory_limit=51
20)])
    except RunTimeError as e:
        print(e)

# Loading pipeline config file
configs =
config_util.get_configs_from_pipeline_file(files['PIPELINE_CONFIG'
])
detection_model =
model_builder.build(model_config=configs['model'],
is_training=False)

# Restoring latest checkpoint
ckpt = tf.compat.v2.train.Checkpoint(model=detection_model)
ckpt.restore(os.path.join(paths['CHECKPOINT_PATH'], 'ckpt-
11')).expect_partial()


# Building a detection model
@tf.function
def detect_fn(image):
    """Build detection model function

        Parameters:
            image (tensorflow.python.framework.ops.EagerTensor):
tensorflow image to detect plate on

        Returns:
            detections (dict, dict_keys(['detection_boxes',
'detection_scores', 'detection_classes', 'raw_detection_boxes',
            'raw_detection_scores', 'detection_multiclass_scores',
'detection_anchor_indices', 'num_detections'])):
```

```
            detected plate"""

    image, shapes = detection_model.preprocess(image)
    prediction_dict = detection_model.predict(image, shapes)
    detections = detection_model.postprocess(prediction_dict,
shapes)
    return detections
```

Pasul cu numărul nouă se concentrează pe detectarea conturului unei plăcuțe de înmatriculare folosind o resursă de tip imagine. Imaginea este mai întâi preporcesată pentru a spori calitatea detecției.

```
img_orig = cv2.imread(IMAGE_PATH)
print('ORIGINAL')
plt.imshow(img_orig)
plt.show()
```

Pasul de redimensionare a imaginii ne ajută să evităm orice probleme referitoare la imaginile cu rezoluție mai mare, asigurându-ne că plăcuța de înmatriculare rămâne în continuare în cadru după redimensionare [47].

Am redimensionat fișierul imagine cu un factor de 2x atât în direcția orizontală, cât și în cea verticală folosind cv2.resize, iar apoi am aplicat o evidențiere a detaliilor.

```
img_resized = cv2.resize(img_orig, None, fx=1.2, fy=1.2,
interpolation=cv2.INTER_CUBIC)
print('IMPROVED RESOLUTION')
plt.imshow(img_resized)
plt.show()

img_detailsEnhanced = cv2.detailEnhance(img_resized, sigma_s=10,
sigma_r=0.15)
print('DETAILS ENHANCED')
plt.imshow(img_detailsEnhanced)
plt.show()

img_BilateralFilterEnhanced =
cv2.bilateralFilter(img_detailsEnhanced, 9, 75, 75)
print('BILATERAL FILTER ENHANCED')
plt.imshow(img_BilateralFilterEnhanced)
plt.show()
```

Convertirea imaginii în tonuri de gri este o practică destul de comună în toate etapele de procesare a imaginii. Acest lucru accelerează alte procese următoare [47].

Convertim imaginea redimensionată în tonuri de gri pentru a optimiza detecția și a reduce numărul de culori prezente în imagine drastic, ceea ce va ajuta, mai departe, în detectarea mai facilă a numerelor de înmatriculare.

```
img_BilateralFilterEnhancedGray =
cv2.cvtColor(img_BilateralFilterEnhanced, cv2.COLOR_BGR2GRAY)
print('BILATERAL GRAY ENHANCED')
plt.imshow(img_BilateralFilterEnhancedGray, cmap='gray')
plt.show()
```

În general, orice imagine va avea informații utile și inutile. În cazul de față, doar plăcuța de înmatriculare reprezintă informație utilă, ce poate fi afectată de zgomot. Utilizăm un filtru bilateral pentru a elimina detaliile nedorite dintr-o imagine [47].

Este continuată preprocesarea aplicând un filtru bilateral, asupra căruia va fi aplicată tehnica de Thresholding Binar și Otsu.

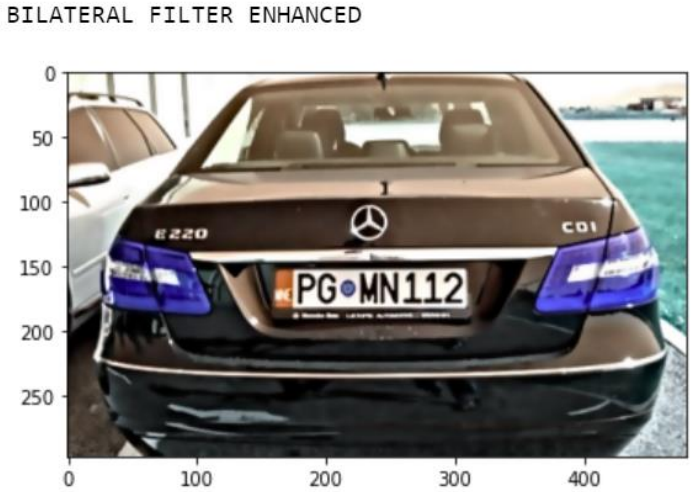
```
img_threshEnhanced =
cv2.threshold(cv2.medianBlur(img_BilateralFilterEnhancedGray, 3),
0, 255, cv2.THRESH_BINARY + cv2.THRESH_OTSU)[1]
print('THRESH 1 ENHANCED')
plt.imshow(img_threshEnhanced, cmap='gray')
plt.show()

img_backtorgb =
cv2.cvtColor(img_threshEnhanced,cv2.COLOR_GRAY2RGB)

image_np = np.array(img_back_to_rgb)

# Detection
input_tensor = tf.convert_to_tensor(np.expand_dims(image_np, 0),
dtype=tf.float32)
detections = detect_fn(input_tensor)
# Filter OCR function
```

Detecția va fi vizualizată pe imaginea originală, fără alte procesări.

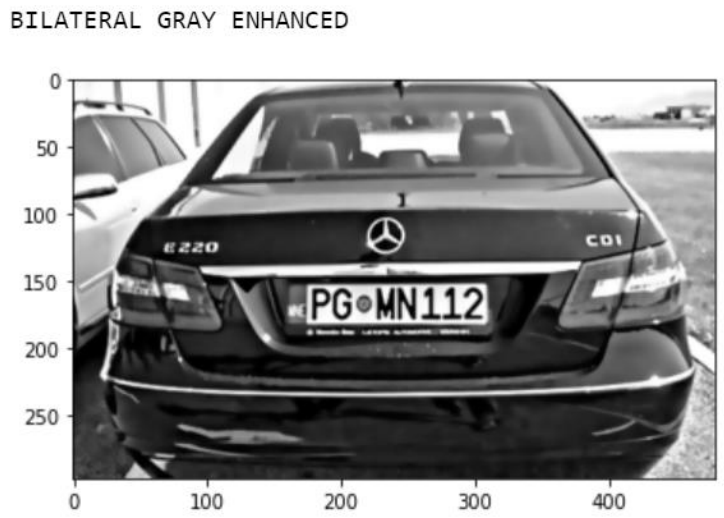
Figură 11 – Imaginea neprocesată

Figură 12 – Imaginea cu rezoluția îmbunătățită

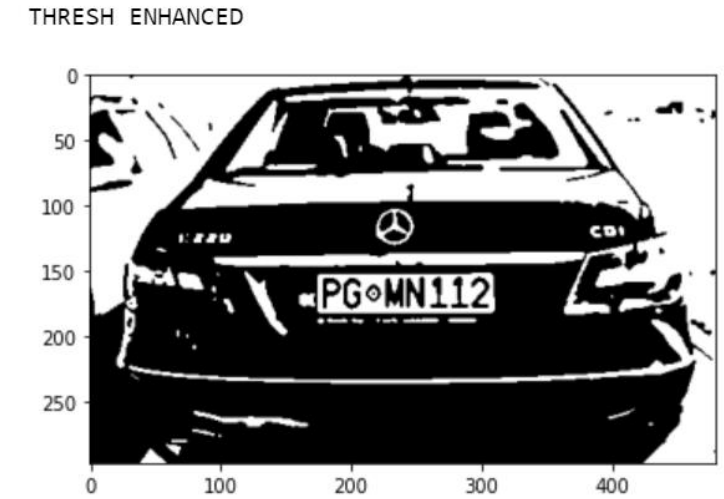
Figură 13 – Imaginea cu detaliile evidențiate



Figură 14 – Imaginea după aplicarea filtrului bilateral



Figură 15 – Imaginea convertită în tonuri de gri



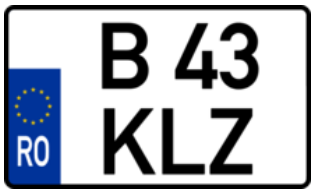
Figură 16 – Imagine după aplicarea thresholding-ului

Figură 17 – Detecția afișată pe imaginea originală

În continuare, se realizează procesul de OCR, mai exact procesarea OCR și filtrarea rezultatelor OCR. Acest lucru se realizează ținând cont de structura nu doar a numerelor de înmatriculare din România, dar și a celor din alte țări. În general, proporțiile plăcuțelor și a textelor de pe plăcuțe sunt asemănătoare de la o țară la alta. Important este ca detecțiile să nu conțină și alte texte ce pot apărea pe o plăcuță, cum ar fi reclame sau codul țării. Aceste detalii nu sunt de interes momentan și pot fi filtrate calculând raportul dintre lățimea și înălțimea textului extras.

Numerele de înmatriculare a mașinilor din România sunt formate dintr-o bandă verticală albastră (banda europeană standard) pe partea stângă a plăcuței care arată cele 12 stele ale Uniunii Europene și codul țării RO, urmată de o suprafață albă, cu scriere neagră, în care apare codul județului și o combinație de două sau trei cifre și trei litere [48].

Figură 18 – Numerele de înmatriculare după 2007 [48]



Figură 19 – Numerele de înmatriculare pentru motociclete după 2007 [48]

Toate plăcuțele emise înainte de 9 mai 2007 au folosit steagul României în locul celor 12 stele europene. Numerele și literele sunt de obicei atribuite aleatoriu, cu excepția cazului în care se plătește o taxă de personalizare [48].

Figură 20 – Numerele de înmatriculare înainte de 2007 [48]

Figură 21 – Numerele de înmatriculare pentru motociclete înainte de 2007 [48]

Plăcuțele de înmatriculare se eliberează pentru fiecare mașină și pentru fiecare proprietar și trebuie returnate dacă mașina este fie vândută, fie casată, deși noul cumpărător are dreptul să solicite utilizarea vechiului număr de înmatriculare [48].

Combinația de numere de înregistrare pentru județe este de tipul JJ NN LLL, unde:

- JJ reprezintă codul județului, astfel: AB, AG, AR, BC, BH, BN, BR, BT, BV, BZ, CJ, CL, CS, CT, CV, DB, DJ, GJ, GL, GR, HD , HR, IF, IL, IS, MH, MM, MS, NT, OT, PH, SB, SJ, SM, SV, TL, TM, TR, VL, VN, VS;
- NN este combinația de numere (de la 01 la 99);
- LLL este combinația de litere (combinațiile nu pot începe cu literele I sau O, deoarece pot fi confundate cu cifrele 1 sau 0, iar litera Q nu poate fi folosită într-o combinație; de asemenea combinațiile III sau OOO sunt interzise) [48].

Combinația de numere de înregistrare pentru municipiul București poate fi de două tipuri: B NN LLL sau B NNN LLL, unde:

- B reprezintă codul Bucureștiului;
- NN sau NNN este combinația de cifre (de la 01 la 99 pentru NN și, respectiv, de la 100 la 999 pentru NNN);
- LLL este o combinație de litere (restricțiile combinației JJ NN LLL sunt păstrate și pentru acest tip de plăcuță de înmatriculare) [48].

Unele instituții au numere speciale, începând cu următoarele litere:

Cod	Instituția
MAI	Ministerul Afacerilor Interne (Poliția, Jandarmeria, Prefecții județelor)
A	Armata
CD	Corp Diplomatic
TC	Transport Consular
FA	Forțele Armate
ALA	Aviație

Tabel 2 – Codurile instituțiilor din România ce au numere de înmatriculare speciale [48]

Numerele temporare pe termen scurt constau din banda europeană, urmată de codul județului și trei până la șase cifre, dintre care prima este întotdeauna zero și a doua este întotdeauna diferită de zero. Toate înscrisurile de pe această plăcuță sunt de culoare roșie [48].

Figură 22 – Numerele temporare pe termen scurt [48]

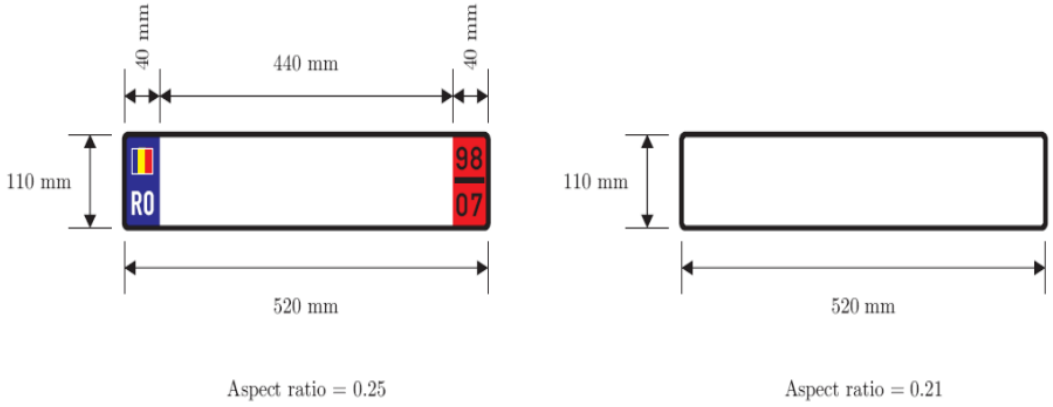
Numerele temporare pe termen lung sunt similare cu numerele pe termen scurt, dar folosesc o inscripție neagră în loc de una roșie, iar numărul nu începe niciodată cu zero [48].

Figură 23 – Numerele temporare pe termen lung [48]

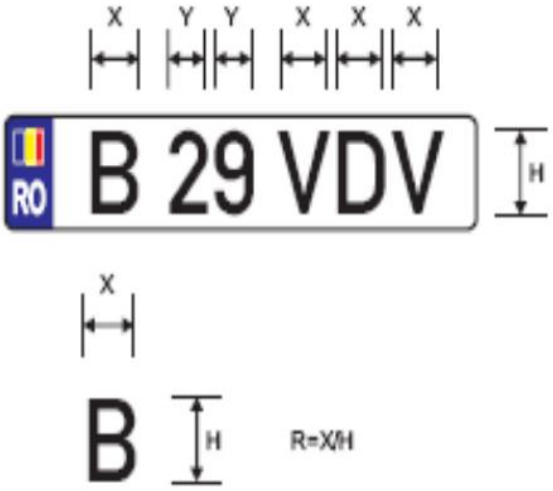
Numerele diplomatice conțin banda europeană urmată de textul albastru. Textul este format dintr-un cod care poate fi CD (Corpul Diplomatic), TC (Transport Consular) sau CO (Consulat), urmat de 6 cifre. Primele trei cifre reprezintă țara sau organizația internațională [48].

Figură 24 – Numerele diplomatice CD (Corp Diplomatic) [48]

Figură 25 – Numerele diplomatice TC (Transport Consular) [48]



Figură 26 – Aspectul numerelor de înmatriculare din România [49]



Figură 27 – Aspectul caracterelor din cadrul numerelor de înmatriculare din România [49]

Astfel, procesul de extragere a caracterelor apelează funcția de filtrare a textului.

```
def filter_text(region, ocr_result, region_threshold) -> list:
    """Filter OCR function

        Parameters:
            region (numpy.ndarray): region of interest for plate
detection
            ocr_result (list): OCR characters
            region_threshold (float): threshold for region to run
OCR over

        Returns:
            plate (list): plate characters"""

    rectangle_size = region.shape[0]*region.shape[1]

    plate = []
    for result in ocr_result:
        length = np.sum(np.subtract(result[0][1], result[0][0]))
        height = np.sum(np.subtract(result[0][2], result[0][1]))

        if length*height / rectangle_size > region_threshold:
            plate.append(result[1])
    return plate

# Thresholds for detection and OCR filtering
detection_threshold = 0.58
region_threshold = 0.6
# OCR function
```

```
def ocr_it(image, detections, detection_threshold,
region_threshold):
    """OCR function

        Parameters:
            image (tensorflow.python.framework.ops.EagerTensor):
tensorflow image to detect plate on
            detections (dict, dict_keys(['detection_boxes',
'detection_scores', 'detection_classes', 'raw_detection_boxes',
            'raw_detection_scores', 'detection_multiclass_scores',
'detection_anchor_indices', 'num_detections'])):
            detected plate:
            detection_threshold (float): threshold for detection
to be considerred successfull
            region_threshold (float): threshold for region to run
OCR over

        Returns:
            text (list): plate characters
            region (tensorflow.python.framework.ops.EagerTensor):
tensorflow image, detected plate region """

    # Scores, boxes and classes above threhold
    scores = list(filter(lambda x: x> detection_threshold,
detections['detection_scores']))
    boxes = detections['detection_boxes'][:len(scores)]
    classes = detections['detection_classes'][:len(scores)]

    # Full image dimensions
    width = image.shape[1]
    height = image.shape[0]

    # Apply ROI filtering and OCR
    for idx, box in enumerate(boxes):
        roi = box*[height, width, height, width]
        region =
image[int(roi[0]):int(roi[2]),int(roi[1]):int(roi[3])]
        reader = easyocr.Reader(['en'])
        ocr_result = reader.readtext(region)

        text = filter_text(region, ocr_result, region_threshold)

        plt.imshow(cv2.cvtColor(region, cv2.COLOR_BGR2RGB))
        plt.show()
        print(text)
        return text, region
```

Salvarea rezultatelor se realizează în fișiere CSV (Comma Separated Values), unde detecțiile sunt grupate pe zile și unde sunt specificate data și ora detecției, fișierul imagine unde este salvată plăcuța și numărul de înmatriculare detectat.

	A	B	C	
1	22-06-19	16-27-27	16-27-27_8fe74502-efd3-11ec-8b33-28dfeb6a105c.jpg	['SAe335c0']
2	22-06-19	16-27-29	16-27-29_9127e9da-efd3-11ec-9695-28dfeb6a105c.jpg	['SA0335c0']
3	22-06-19	16-27-31	16-27-31_9267b8c2-efd3-11ec-b507-28dfeb6a105c.jpg	['SA0335c0']
4	22-06-19	16-27-33	16-27-33_93a8c906-efd3-11ec-bbb3-28dfeb6a105c.jpg	['SA0335c0']
5	22-06-19	16-27-35	16-27-35_94f0c3a0-efd3-11ec-802c-28dfeb6a105c.jpg	['SAe335c0']
6	22-06-19	16-27-37	16-27-37_9631d6e6-efd3-11ec-ad06-28dfeb6a105c.jpg	['SA0335c0']
7	22-06-19	16-27-40	16-27-40_977dba3e-efd3-11ec-a8df-28dfeb6a105c.jpg	['SAe335c0l']
8	22-06-19	16-27-42	16-27-42_98db34bb-efd3-11ec-815b-28dfeb6a105c.jpg	['SA0335C0']
9	22-06-19	16-27-45	16-27-45_9a8dd9da-efd3-11ec-97d4-28dfeb6a105c.jpg	['SAe335c0']
10	22-06-19	16-27-47	16-27-47_9bceaf39-efd3-11ec-9f51-28dfeb6a105c.jpg	['SAe335c0']
11	22-06-19	16-27-49	16-27-49_9d156c50-efd3-11ec-ba09-28dfeb6a105c.jpg	['SA0335c0']
12	22-06-19	16-27-59	16-27-59_a35f5bbf-efd3-11ec-bbc4-28dfeb6a105c.jpg	['SA0335c0']

Figură 28 – Salvarea rezultatelor preliminare în fișiere CSV

Imaginile preluate sunt salvate ca fișiere JPG (JPEG - Joint Photographic Experts Group), iar numele acestora conține ora detecției și un identificator generat unic. Imaginile sunt grupate în directoare conform datei preluării detecției.

Figură 29 – Salvarea fișierelor imagine

După definirea funcției de salvare a rezultatelor, urmează blocul de cod în care se realizează detecția numerelor de înmatriculare în timp real, de la o resursă de tip cameră web. Acest bloc de cod grupează toate funcțiile definite anterior.

Toate acestea fiind realizate, urmează procesarea rezultatelor. Funcția getDatafromCsv() preia datele din toate fișierele CSV salvate. Mai departe, aceste rezultate sunt trecute prin funcția redundancyFunction, care grupează detecțiile în funcție de minutul

în care au fost realizate, pentru a le procesa mai departe, prin funcția processPlate, care returnează o singură detecție combinată din toate detecțiile dintr-un minut.

De exemplu, din detecțiile din Tabelul 4, va fi returnat rezultatul următor:

22-06-20	**19-54-28**	**19-54-28_a5ad93e5-f0b9-11ec-8dc7-28dfeb6a105c.jpg**	**['SV13CBC']**

Tabel 3 – Exemplu de rezultat al funcției redundancyFunction

Figură 30 – Fișierul imagine menționat în Tabelul 3

22-06-20	**19-54-25**	**19-54-25_a46d578c-f0b9-11ec-8b41-28dfeb6a105c.jpg**	**['unreadable']**
22-06-20	19-54-28	19-54-28_a5ad93e5-f0b9-11ec-8dc7-28dfeb6a105c.jpg	['SV 13CBC']
22-06-20	19-54-30	19-54-30_a6eccb99-f0b9-11ec-a26f-28dfeb6a105c.jpg	['SV 13CBC']
22-06-20	19-54-32	19-54-32_a82ea580-f0b9-11ec-b305-28dfeb6a105c.jpg	['SV 13CBC']
22-06-20	19-54-34	19-54-34_a96b93ce-f0b9-11ec-bc40-28dfeb6a105c.jpg	['SV 13CBC']
22-06-20	19-54-36	19-54-36_aab6fa8d-f0b9-11ec-ac19-28dfeb6a105c.jpg	['SV 13CBC}']
22-06-20	19-54-38	19-54-38_abf85db3-f0b9-11ec-bb3d-28dfeb6a105c.jpg	['SV 13CBC']
22-06-20	19-54-44	19-54-44_afa90900-f0b9-11ec-ac9e-28dfeb6a105c.jpg	['FV 13CBC']
22-06-20	19-54-46	19-54-46_b0e8e227-f0b9-11ec-a00f-28dfeb6a105c.jpg	['SV 13CBC']
22-06-20	19-54-49	19-54-49_b22e4f53-f0b9-11ec-9f8a-28dfeb6a105c.jpg	['FSV 13CBC']
22-06-20	19-54-51	19-54-51_b373bda3-f0b9-11ec-954d-28dfeb6a105c.jpg	['unreadable']
22-06-20	19-54-53	19-54-53_b4b7f18f-f0b9-11ec-a584-28dfeb6a105c.jpg	['unreadable']
22-06-20	19-54-55	19-54-55_b5f4fb26-f0b9-11ec-8c82-28dfeb6a105c.jpg	['SV 13CBC']
22-06-20	19-54-57	19-54-57_b730e730-f0b9-11ec-921e-28dfeb6a105c.jpg	['SV 13CBC']
22-06-20	19-54-59	19-54-59_b872a218-f0b9-11ec-a9f0-28dfeb6a105c.jpg	['SV 13CBC']

Tabel 4 – Input-urile funcției redundancyFunction pentru rezultatul din Tabelul 3

O altă funcție necesară este cea de trimitere de alerte via email în momentul în care este considerat că o mașină urmărește utilizatorul.

```
smtp_server = "smtp.mail.yahoo.com"
port = 587  # For starttls
sender = 'anprthesis@yahoo.com'
receiver = 'anprthesis@gmail.com'
password = 'wamxfatshsvkanyb'


# Create a secure SSL context
context = ssl.create_default_context()



def sendAlert(followerType, followerPlate):
    """Send mail alerts function.

    Parameters:
        followerType (str): follower type
        followerPlate (str): follower plate

    Returns: N/A"""

    # Try to log in to server and send email
    try:
        server = smtplib.SMTP(smtp_server,port)
        server.ehlo()
        server.starttls(context=context) # Secure the connection
        server.ehlo()
        server.login(sender, password)

        emailMsg = EmailMessage()

        nowMoment = datetime.now()
        date_string = nowMoment.strftime('%y-%m-%d')
        time_string = nowMoment.strftime('%H-%M-%S')

        msg = f'You are being followed by {followerPlate}, date
{date_string}, time {time_string}.\nType of follower:
{followerType}.'

        emailMsg.set_content(msg)

        emailMsg['Subject'] = f'Dangerous car behind!'
        emailMsg['From'] = sender
        emailMsg['To'] = receiver
```

```
        server.send_message(emailMsg)

    except Exception as e:
        print(e)

    finally:
        server.quit()
```

Funcțiile menționate până acum, împreună cu un proces de curățare a fișierelor și un scurt algoritm de determinare a unui urmăritor sunt folosite în pasul 13, în două thread-uri diferite, pentru procesarea în paralel a tuturor acestor operații.

```
# File cleanup
src = 'RealTimeDetections\\NoBlanks'
trg = 'RealTimeDetections\\ForDetections'
deleteProcessed = 'RealTimeDetections\\Processed'
deleteNoBlanks = 'RealTimeDetections\\NoBlanks'

sourceFiles = os.listdir(src)
targetFiles = os.listdir(trg)
deleteProcessedFiles = os.listdir(deleteProcessed)
deleteNoBlanksFiles = os.listdir(deleteNoBlanks)

for fname in targetFiles:
    if (isfile(join(trg, fname)) and fname.endswith('.csv')):
        os.remove(os.path.join(trg, fname))

# Copying files used for alerts to their correct directory
for fname in sourceFiles:
    shutil.copy2(os.path.join(src, fname), trg)


for fname in deleteNoBlanksFiles:
    if (isfile(join(deleteNoBlanks, fname)) and
fname.endswith('.csv')):
        os.remove(os.path.join(deleteNoBlanks, fname))
for fname in deleteProcessedFiles:
    if (isfile(join(deleteProcessed, fname)) and
fname.endswith('.csv')):
        os.remove(os.path.join(deleteProcessed, fname))


def task1():
    """
    Task 1 for parallel processing. Real Time detections and
alerts.
    """

    # Computing a list of previous detections
```

```
    csvForDetectionsfiles = [f for f in
listdir('RealTimeDetections\\ForDetections') if
isfile(join('RealTimeDetections\\ForDetections', f)) and
f.endswith('.csv')]

    previousDetections = []
    for elemCsvForDetections in csvForDetectionsfiles:

        detections =
pd.read_csv(os.path.join('RealTimeDetections\\ForDetections',
elemCsvForDetections), header=None, usecols=[3])
        listDetections = detections.values.tolist()
        listDetectionsNew = [elem[0] for elem in listDetections]
        listDetectionsNewNew = [elem[2:-2] for elem in
listDetectionsNew]
        for elem in listDetectionsNewNew:
            previousDetections.append(elem)

    # Real Time Detections from Webcam
    cap = cv2.VideoCapture(0)
    width = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
    height = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))

    nowDetections = []
    while cap.isOpened():
        ret, frame = cap.read()

        # Preprocessing image
        img_resized = cv2.resize(frame, None, fx=1.2, fy=1.2,
interpolation=cv2.INTER_CUBIC)
        img_detailsEnhanced = cv2.detailEnhance(img_resized,
sigma_s=10, sigma_r=0.15)
        img_BilateralFilterEnhanced =
cv2.bilateralFilter(img_detailsEnhanced, 9, 75, 75)
        img_BilateralFilterEnhancedGray =
cv2.cvtColor(img_BilateralFilterEnhanced, cv2.COLOR_BGR2GRAY)
        img_threshEnhanced =
cv2.threshold(cv2.medianBlur(img_BilateralFilterEnhancedGray, 3),
0, 255, cv2.THRESH_BINARY + cv2.THRESH_OTSU)[1]
        img_backtorgb =
cv2.cvtColor(img_threshEnhanced,cv2.COLOR_GRAY2RGB)

        image_np = np.array(frame)
        image_np_preprocessed = np.array(img_backtorgb)

        input_tensor =
tf.convert_to_tensor(np.expand_dims(image_np_preprocessed, 0),
dtype=tf.float32)
        detections = detect_fn(input_tensor)
```

```
        num_detections = int(detections.pop('num_detections'))
        detections = {key: value[0, :num_detections].numpy()
                      for key, value in detections.items()}
        detections['num_detections'] = num_detections

        # Detection_classes should be ints.
        detections['detection_classes'] =
detections['detection_classes'].astype(np.int64)

        label_id_offset = 1
        image_np_with_detections = image_np.copy()


viz_utils.visualize_boxes_and_labels_on_image_array(image_np_with_
detections,detections['detection_boxes'],detections['detection_cla
sses']+label_id_offset,detections['detection_scores'],category_ind
ex,use_normalized_coordinates=True,max_boxes_to_draw=5,min_score_t
hresh=.8,agnostic_mode=False)

        try:
            text, region = ocr_it(image_np_with_detections,
detections, detection_threshold, region_threshold)
            save_results(text, region)

            toCheck = text[0]
            toCheck = toCheck.replace(' ', '')
            toCheck = toCheck.upper()

            # Sending alerts
            if ((toCheck in previousDetections) or
(nowDetections.count(toCheck) > 1)):
                nowDetections.append(toCheck)
                if ((toCheck in previousDetections) and
(nowDetections.count(toCheck) > 1)):
                    typeOfFollowing = 'detected previously and
following you now'
                    print(f'Danger, {typeOfFollowing}.')
                    sendAlert(typeOfFollowing, toCheck)
                elif (toCheck in previousDetections):
                    typeOfFollowing = 'detected in a previous
moment'
                    print(f'Danger, {typeOfFollowing}.')
                    sendAlert(typeOfFollowing, toCheck)
                else:
                    typeOfFollowinf = 'detected for too long in
this moment'
                    print(f'Danger, {typeOfFollowing}.')
                    sendAlert(typeOfFollowing, toCheck)
            else:
                print('Safe')
```

```
        except:
            pass

        cv2.imshow('object detection',
cv2.resize(image_np_with_detections, (800, 600)))

        if cv2.waitKey(10) & 0xFF == ord('q'):
            cap.release()
            cv2.destroyAllWindows()
            break


def task2():
    """
    Task 2 for parallel processing. Cleanup of files, processing
past detections.
    """

    csvfiles = [f for f in listdir('RealTimeDetections') if
isfile(join('RealTimeDetections', f)) and f.endswith('.csv')]
    # First one doesn't conform to the template
    # csvfiles = csvfiles[2:]
    # print(csvfiles)


    for elem in csvfiles:
        csvReturns = []
        redReturns = []
        finalPlate = ''

        [csvListDateReturn, csvListTimeReturn,
csvListFileNameReturn, csvListPlateReturn] = getDataFromCsv(elem)
        csvReturns.extend([csvListDateReturn, csvListTimeReturn,
csvListFileNameReturn, csvListPlateReturn])

        #print(csvReturns[0])
        #print()
        #print(csvReturns[1])
        #print()
        #print(csvReturns[2])
        #print()
        #print(csvReturns[3])
        #print('----------------------------------------')

        [redListDateReturn, redListTimeReturn,
redListFileNameReturn, redListPlateReturn] =
redundancyFunction(csvReturns[0], csvReturns[1], csvReturns[2],
csvReturns[3])
```

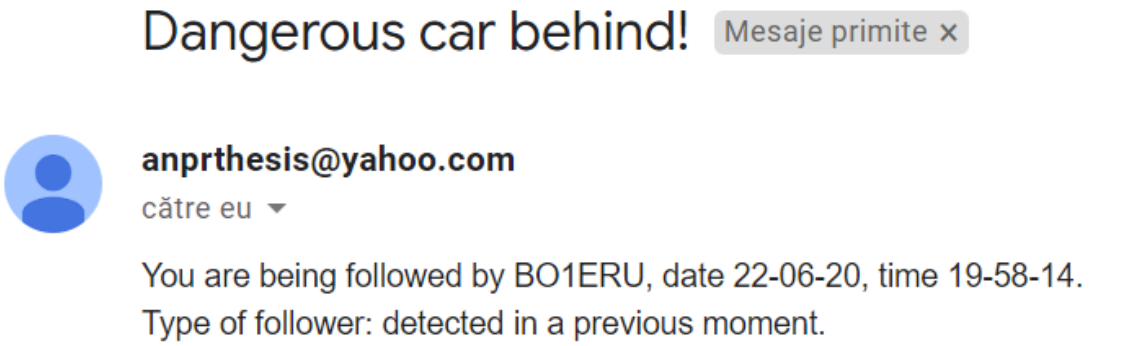
```
        redReturns.extend([redListDateReturn, redListTimeReturn,
redListFileNameReturn, redListPlateReturn])

        #print(redReturns[0])
        #print()
        #print(redReturns[1])
        #print()
        #print(redReturns[2])
        #print()
        #print(redReturns[3])
        #print('-------------------------------------')

        for i, platesToProc in enumerate (redReturns[3]):
            finalPlate = processPlate(platesToProc,
redReturns[1][i], redReturns[0][i], redReturns[2][i])

            f = open(join('RealTimeDetections\\Processed',
f'Processed_{finalPlate[1]}.csv'), 'a')

            # create the csv writer
            writer = csv.writer(f)

            row = [finalPlate[1], finalPlate[2], finalPlate[3],
finalPlate[0]]
            # write a row to the csv file
            writer.writerow(row)

            # close the file
            f.close()

        csvProcessedfiles = [f for f in
listdir('RealTimeDetections\\Processed') if
isfile(join('RealTimeDetections\\Processed', f)) and
f.endswith('.csv')]

        for elem in csvProcessedfiles:
            findIndex = elem.find('.csv')
            elemOutput = elem[:findIndex]
            elemOutput = elemOutput + "_NoBlanks.csv"

        with open(join('RealTimeDetections\\Processed', elem),
'r') as inputFile, open(join('RealTimeDetections\\NoBlanks',
elemOutput), 'w', newline='') as outputFile:
            writer = csv.writer(outputFile)
            for row in csv.reader(inputFile):
                if any(field.strip() for field in row):
                    writer.writerow(row)


start_time = perf_counter()
```

```
# Create two new threads
t1 = Thread(target=task1)
t2 = Thread(target=task2)

# Start the threads
t1.start()
t2.start()

# Wait for the threads to complete
t1.join()
t2.join()

end_time = perf_counter()

print(f'It took {end_time- start_time: 0.2f} second(s) to
complete.')
# Freeze script from TFOD
```

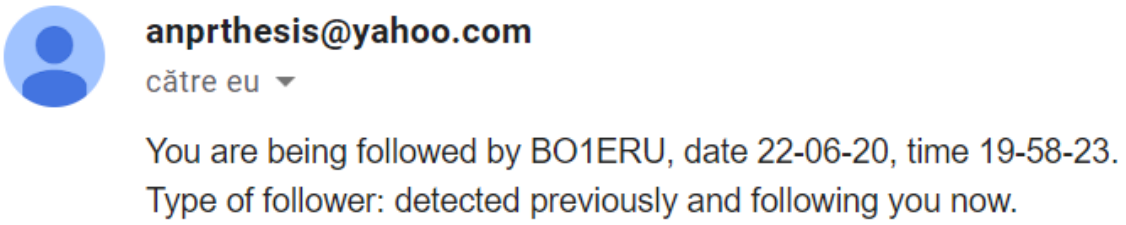
Momentan, detecția urmăritorilor se face în următorul fel. Dacă numărul de înmatriculare a fost detectat și într-o zi anterioară, atunci mașina căreia îi aparține numărul de înmatriculare este considerată periculoasă. De asemenea, dacă în momentul actual de detecție a fost detectată de mai mult de o singură dată, atunci este considerată a fi o mașină periculoasă. Decizia a fost luată pentru a putea demonstra funcționalitatea sistemului, dar, într-un cadru real, în trafic, parametrii trebuie modificați.


```
['BLO654BP']
22-06-21
09-22-07
Danger, detected in a previous moment.
```

Figură 31 – Atenționare urmăritor în cadrul programului



Figură 32 – Exemplu de atenționare urmăritor via mail, mașină detectată anterior



Figură 33 – Exemplu de atenționare urmăritor via email, mașină detectată anterior și care urmărește utilizatorul în momentul actual

De asemenea, pentru a nu fi nevoiți să oferim ca input de la tastatură de fiecare dată în momentul testării camera video pe care o folosim, adresa de email și placa grafică ce este folosită, acestea au fost reținute în cadrul codului. Ulterior, pot fi modificate în input-uri de la tastatură sau schimbate în interiorul codului.

În final, sunt folosite script-uri de la Tensorflow pentru a exporta modelul antrenat sub forma de TFJS și TFLite. TFJS este structura potrivită aplicațiilor în browser, folosită de obicei cu Javascript sau Typescript. TFlite este folosit, în general, cu Python, JAVA, C++, etc., potrivit pentru aplicații mobile și device-uri IoT (Internet of Things).

```
FREEZE_SCRIPT = os.path.join(paths['APIMODEL_PATH'], 'research',
'object_detection', 'exporter_main_v2.py ')

# Generating freezing command
command = "python {} --input_type=image_tensor --
pipeline_config_path={} --trained_checkpoint_dir={} --
output_directory={}".format(FREEZE_SCRIPT
,files['PIPELINE_CONFIG'], paths['CHECKPOINT_PATH'],
paths['OUTPUT_PATH'])

# Generating conversion to TFJS command
```

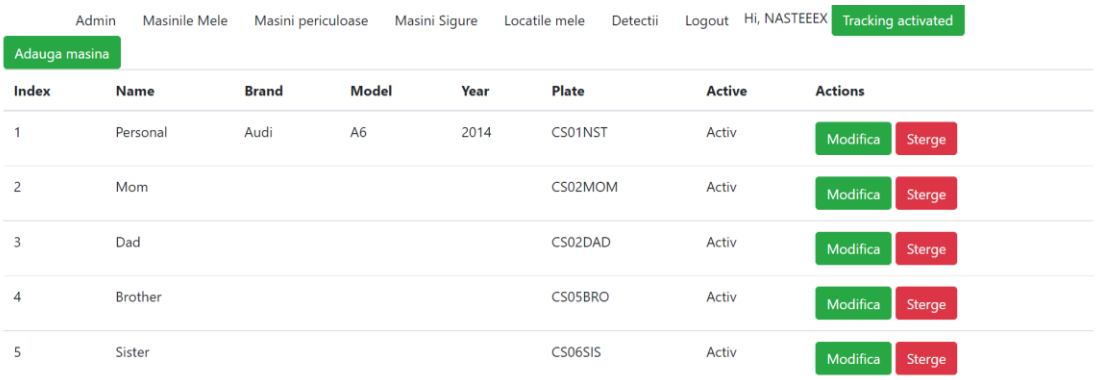
```
command = "tensorflowjs_converter --input_format=tf_saved_model --
output_node_names='detection_boxes,detection_classes,detection_fea
tures,detection_multiclass_scores,detection_scores,num_detections,
raw_detection_boxes,raw_detection_scores' --
output_format=tfjs_graph_model --signature_name=serving_default {}
{}".format(os.path.join(paths['OUTPUT_PATH'], 'saved_model'),
paths['TFJS_PATH'])

# TFLite script from TFOD
TFLITE_SCRIPT = os.path.join(paths['APIMODEL_PATH'], 'research',
'object_detection', 'export_tflite_graph_tf2.py ')

# Generating conversion to TFLite command
command = "python {} --pipeline_config_path={} --
trained_checkpoint_dir={} --
output_directory={}".format(TFLITE_SCRIPT
,files['PIPELINE_CONFIG'], paths['CHECKPOINT_PATH'],
paths['TFLITE_PATH'])

# Creating path and file for TFLite model
FROZEN_TFLITE_PATH = os.path.join(paths['TFLITE_PATH'],
'saved_model')
TFLITE_MODEL = os.path.join(paths['TFLITE_PATH'], 'saved_model',
'detect.tflite')

# Updating conversion to TFLite command
command = "tflite_convert \
--saved_model_dir={} \
--output_file={} \
--input_shapes=1,300,300,3 \
--input_arrays=normalized_input_image_tensor \
--
output_arrays='TFLite_Detection_PostProcess','TFLite_Detection_Pos
tProcess:1','TFLite_Detection_PostProcess:2','TFLite_Detection_Pos
tProcess:3' \
--inference_type=FLOAT \
--allow_custom_ops".format(FROZEN_TFLITE_PATH, TFLITE_MODEL, )
```

De asemenea, pentru vizualizarea rezultatelor într-o interfață mai atractivă, am creat o aplicație Django minimală folosind Django Framework, programare orientată pe obiecte în Python, elemente de CSS și HTML.

Pagina principală a aplicației web prezintă formularul de logare și anteturile paginilor ce pot fi accesate, împreuna cu butonul pentru pornirea/oprirea detecțiilor (funcționalitate neimplementată).

Figură 34 – Meniu aplicație web

Utilizatorul poate să vizualizeze tabelar mașinile acestuia, mașinile marcate ca fiind sigure (Figura 35), mașinile marcate ca reprezentând un pericol, locațiile acestuia (Figura 36) și detecțiile numerelor de înmatriculare (Figura 37). Fiecare rând al unui tabel poate fi modificat sau dezactivat.

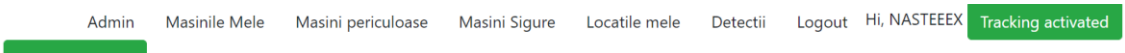


Index	Name	Brand	Model	Year	Plate	Active	Actions
1	Personal	Audi	A6	2014	CS01NST	Activ	Modifica Sterge
2	Mom				CS02MOM	Activ	Modifica Sterge
3	Dad				CS02DAD	Activ	Modifica Sterge
4	Brother				CS05BRO	Activ	Modifica Sterge
5	Sister				CS06SIS	Activ	Modifica Sterge

Figură 35 – Mașini marcate ca fiind sigure în aplicația web

Index	Name	Coordinates	Last Accessed	Marked As	Active	Actions
1	Home	45.432832 / N 45° 25' 58.195'' --- 22.239642 / E 22° 14' 22.709''	22-05-28	Safe	Activ	Modifica Sterge
2	School			Not marked	Activ	Modifica Sterge
3	Work			Not marked	Activ	Modifica Sterge

Figură 36 – Locații utilizator aplicație web



Index	Plate	Date	Time	File	Marked As	Active	Actions
1	B OIERU	22-05-26	20-55-33	20-55-33_0a026e95-dd1d-11ec-b875-28dfeb6a105c.jpg	Safe	Activ	Modifica Sterge
2	MHIZDE1433	22-05-25	17-41-54	17-41-54_d2244524-dc38-11ec-8f03-28dfeb6a105c.jpg	Safe	Activ	Modifica Sterge
3	BLV654BP	22-05-25	17-42-23	17-42-23_e384acf8-dc38-11ec-b1d7-28dfeb6a105c.jpg	Safe	Activ	Modifica Sterge
4	BOIRU	22-05-25	17-43-02	17-43-02_fa7dcc48-dc38-11ec-a797-28dfeb6a105c.jpg	Not marked	Activ	Modifica Sterge
5	TVVOOOO7	22-05-26	20-54-41	20-54-41_eb3481cf-dd1c-11ec-a384-28dfeb6a105c.jpg	Safe	Activ	Modifica Sterge

Figură 37 – Detecții aplicație web

6. EVALUAREA REZULTATELOR

6.1 TESTARE

Modelul antrenat a fost testat pe un set de date relativ mediu raportat la mărimea setului de date pentru antrenare. Rezultatul returnat a fost o acuratețe a detecției plăcuțelor de înmatriculare medie de 68.17647059%. Acest procent este satisfăcător, în sensul în care anumite imagini alese pentru testare reprezentau o provocare pentru sistem sau erau considerate chiar a fi imposibile pentru o detecție mulțumitoare, fiind imagini de o calitate nemulțumitoare, realizate de la distanțe relativ mari și din unghiuri diverse.

Index	Fișier	Acuratețe detecție
1	Cars0.png	71
2	Cars1.png	91
3	Cars2.png	61
4	Cars3.png	91
5	Cars4.png	91
6	Cars5.png	0
7	Cars6.png	90
8	Cars7.png	14
9	Cars8.png	85
10	Cars9.png	70
11	Cars10.png	91
12	Carsa.png	90
13	Carsb.png	0
14	Carsc.png	91
15	TEST4.jpg	57
16	TEST9.jpg	61
17	TEST11.jpg	68
18	TEST12.jpg	76
19	TEST13.jpg	82
20	TEST14.jpg	84
21	TEST15.jpg	72
22	TEST16.jpg	76
23	TEST17.jpg	46
24	TEST18.jpg	51
25	TEST19.jpg	72
26	TEST20.jpg	86
27	TEST21.jpg	39

28	TEST22.jpg	82
29	TEST23.jpg	79
30	TEST24.jpg	85
31	TEST25.jpg	61
32	TEST26.jpg	57
33	TEST27.jpg	62
34	TEST28.jpg	86
		68.17647059

Tabel 5 – Acuratețea detecțiilor plăcuțelor de înmatriculare realizate pe setul de date de testare

De asemenea, am calculat și precizia detecției textului numerelor de înmatriculare în timp real. Folosind detecțiile din 151 de minute, sistemul returnează 151 de numere de înmatriculare. Calculând procentul dintre caracterele detectate corect și cele detectate greșit, sistemul are o rată de succes de 53.639%. Dificultăți, precum era de așteptat, au fost întâmpinate la diferențierea între caractere asemănătoare, precum I și 1, O și 0, Z si 2 sau B și 3. De asemenea, unele plăcuțe de înmatricularea conțineau dovada achitării unei taxe de drum sub forma unui sticker. Acest sticker, de foarte multe ori, este interpretat ca fiind un caracter (uzual caracterul O), inserat în mijlocul textului, deplasând celelalte caractere la dreapta. Astfel, deși, în unele cazuri, era doar un caracter inserat în plus, toate celelalte caractere ce îi urmau erau considerate greșite, returnând o precizie scăzută.

Exemple de greșeli uzuale:

- Numărul de înmatriculare MH12DE1433 detectat ca fiind MHIZDE1433,
- Numărul de înmatriculare B01ERU detectat ca fiind BOIERU,
- Numărul de înmatriculare BL654BP interpretat ca fiind BLO654BP,

6.2 EVALUARE

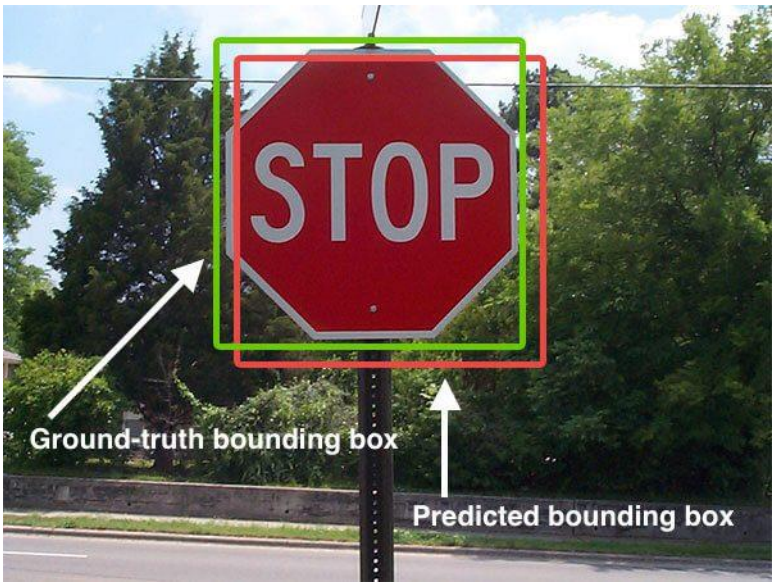
Una dintre provocările cele mai mari privind realizarea unui sistem ANPR implică găsirea unui set de date ce poate fi utilizat pentru a antrena un model de învățare automată [50].

Adesea, nu modelul instruit este valoros, ci setul de date pe care o anumită companie l-a organizat. Seturi de date ANPR de dimensiuni largi și robuste pentru antrenare și testare sunt dificil de obținut, deoarece aceste seturi de date conțin informații sensibile, cu caracter personal, iar diversele companii și entități guvernamentale ce dețin astfel de seturi de date nu doresc a le împărtăși din motive de competivitate [51]. Un set de date extins este punctul cheie către o evaluare completă.

Intersecția peste Uniune (în Engleză, Intersection over Union – IoU) este o metrică de evaluare utilizată pentru a măsura acuratețea unui detector de obiecte pe un anumit set de date. Orice algoritm care furnizează casete de delimitare prezise ca ieșire poate fi evaluat folosind IoU [52].

Pentru a aplica IoU pentru a evalua un detector de obiecte, avem nevoie de:

- Casetele de delimitare a adevărului de bază (adică acele casete de delimitare etichetate manual din setul de testare care specifică unde se află în imagine obiectul căutat),
- Casetele de delimitare prezise de către modelul nostru.



Figură 38 - Un exemplu de detectare a unui indicator de circulație de oprire într-o imagine. Caseta de delimitare prezisă este desenată cu roșu, în timp ce caseta de delimitare a adevărului de bază este desenată cu verde [52].

Pentru a calcula IoU, este de ajuns să împărțim suprafața de suprapunere la suprafața de uniune.

$$IoU = \frac{Suprafața\ de\ suprapunere}{Suprafața\ de\ uniune} \quad (1)$$

Figură 39 – Suprafața de suprapunere [52]

Figură 40 – Suprafața de uniune [52]

Fiecare IoU are mapat precizia medie și factorul mediu de Recall. Precizia măsoară cât de exacte sunt predicțiile realizate, adică procentajul previziunilor corecte. Recall-ul măsoară cât de bine au fost determinate predicțiile pozitive [53]. Formulele pentru Precizie și Recall:

$$TP = Adevărat\ pozitiv \tag{2}$$

$$TN = Adevărat\ negativ \tag{3}$$

$$FP = Fals\ pozitiv \tag{4}$$

$$FN = Fals\ negativ \tag{5}$$

$$Precizie = \frac{TP}{TP+FP} \tag{6}$$

$$Recall = \frac{TP}{TP+FN} \tag{7}$$

IoU	area	maxDets	Average Precision
0.50:0.95	all	100	0.533
0.5	all	100	0.901
0.75	all	100	0.535
0.50:0.95	small	100	0.134
0.50:0.95	medium	100	0.66
0.50:0.95	large	100	0.751

Tabel 6 – Precizia medie în funcție de IoU

Se poate observa, din Tabelul 6, că modelul a returnat o acuratețe mai mare folosind o arie de detecție extinsă, la un IoU de 0.5.

IoU	area	maxDets	Average Recall
0.50:0.95	all	1	0.6
0.50:0.95	all	10	0.6
0.50:0.95	all	100	0.6
0.50:0.95	small	100	0.133
0.50:0.95	medium	100	0.75
0.50:0.95	large	100	0.8

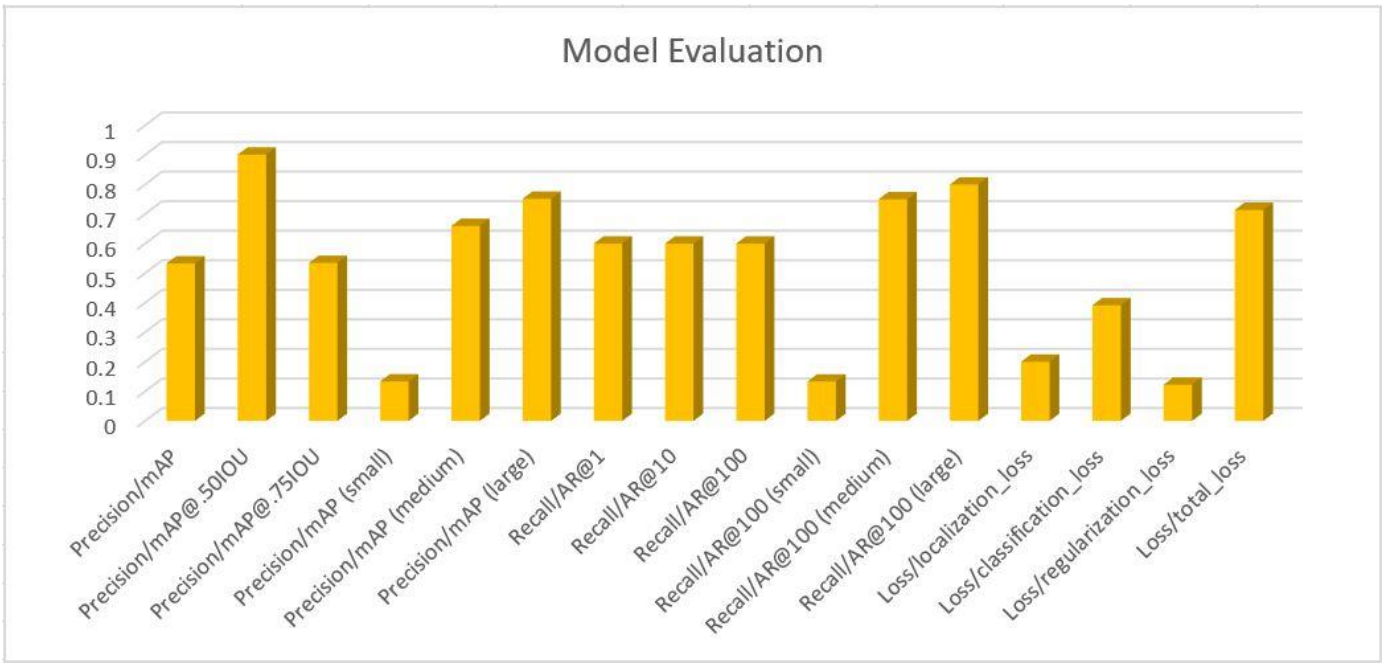
Tabel 7 – Recall-ul mediu în funcție de IoU

La un Recall identic pentru fiecare arie de detecție, Recall-ul mediu a returnat rezultatele cele mai mari pentru aria de detecție totală, rezultate ce pot fi vizualizate în Tabelul 7.

În tabelul 8 sunt prezentate aceste evaluări anterioare, împreună cu pierderile modelului, iar reprezentarea grafică a acestora se poate urmări în Figura 37.

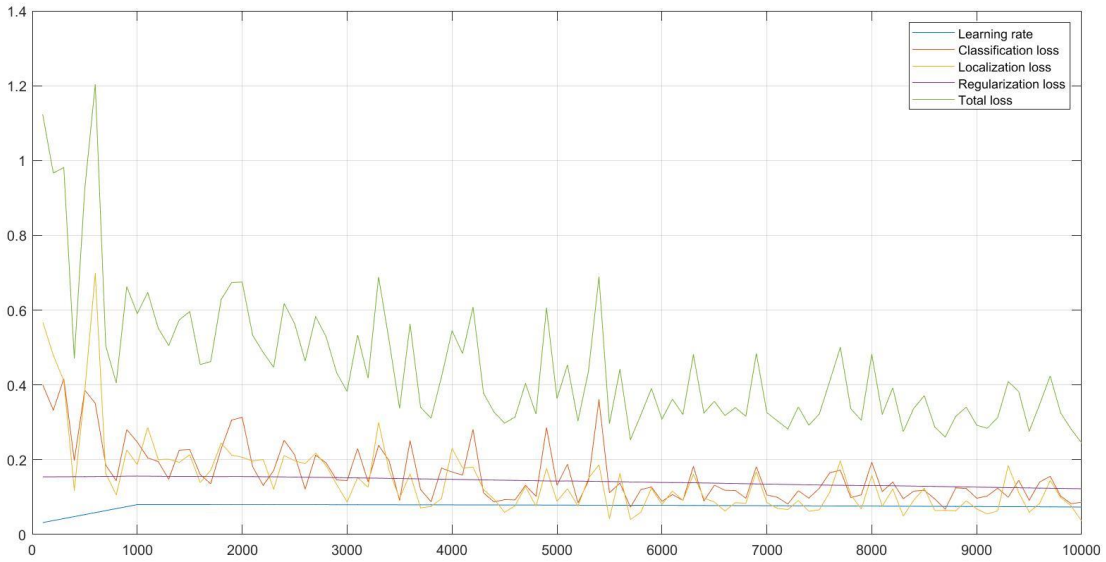
Step	10000
DetectionBoxes_Precision/mAP	0.53269
DetectionBoxes_Precision/mAP@.50IOU	0.90099
DetectionBoxes_Precision/mAP@.75IOU	0.534653
DetectionBoxes_Precision/mAP (small)	0.133663
DetectionBoxes_Precision/mAP (medium)	0.659736
DetectionBoxes_Precision/mAP (large)	0.751485
DetectionBoxes_Recall/AR@1	0.6
DetectionBoxes_Recall/AR@10	0.6
DetectionBoxes_Recall/AR@100	0.6
DetectionBoxes_Recall/AR@100 (small)	0.133333
DetectionBoxes_Recall/AR@100 (medium)	0.75
DetectionBoxes_Recall/AR@100 (large)	0.8
Loss/localization_loss	0.200382
Loss/classification_loss	0.391242
Loss/regularization_loss	0.122457
Loss/total_loss	0.714081

Tabel 8 – Precizia medie, recall-ul mediu și pierderile în funcție de IoU, la pasul 10000

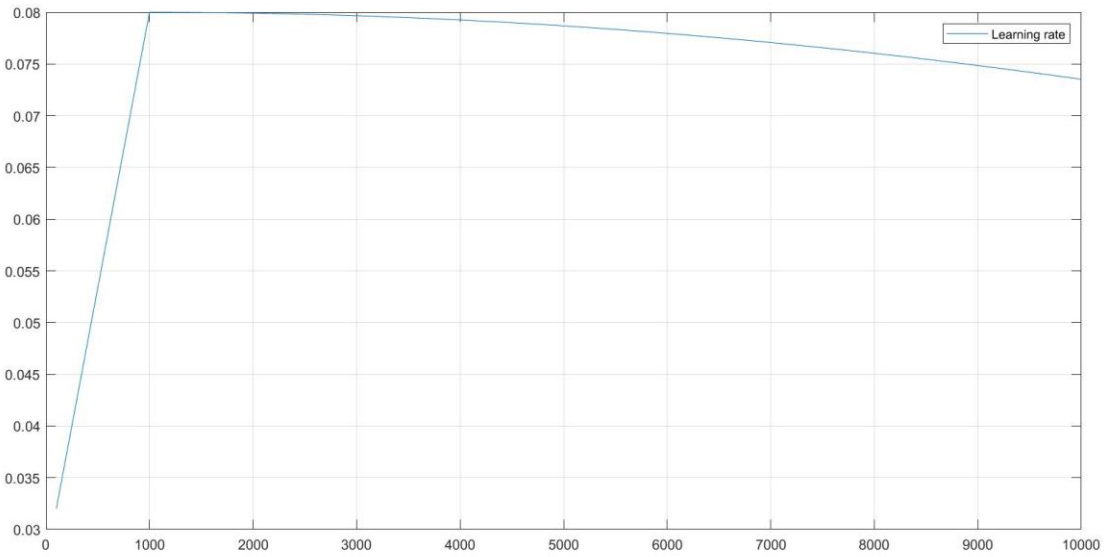


Figură 41 - Precizia medie, recall-ul mediu și pierderile în funcție de IoU, la pasul 10000

Figura 38 prezintă graficul MATLAB al tuturor pierderilor, pe parcursul celor 10000 de pași de antrenare.

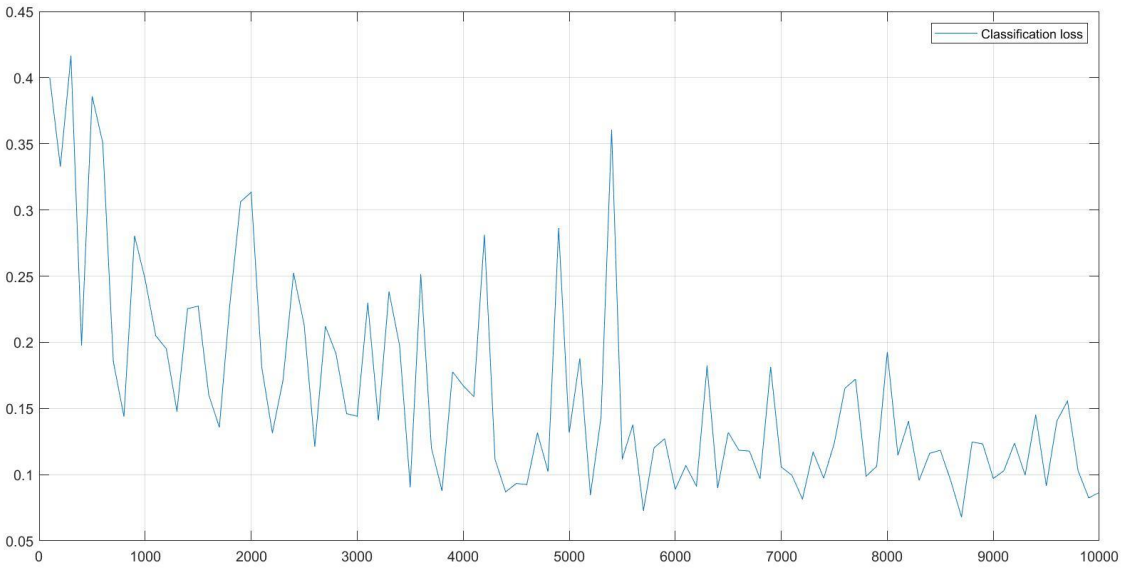


Figură 42 – Rata de învățare, Pierderea de clasificare, Pierderea de localizare, Pierderea de regularizare și Pierderea totală, în funcție de pasul de antrenare



Figură 43 - Rata de învățare în funcție de pasul de antrenare

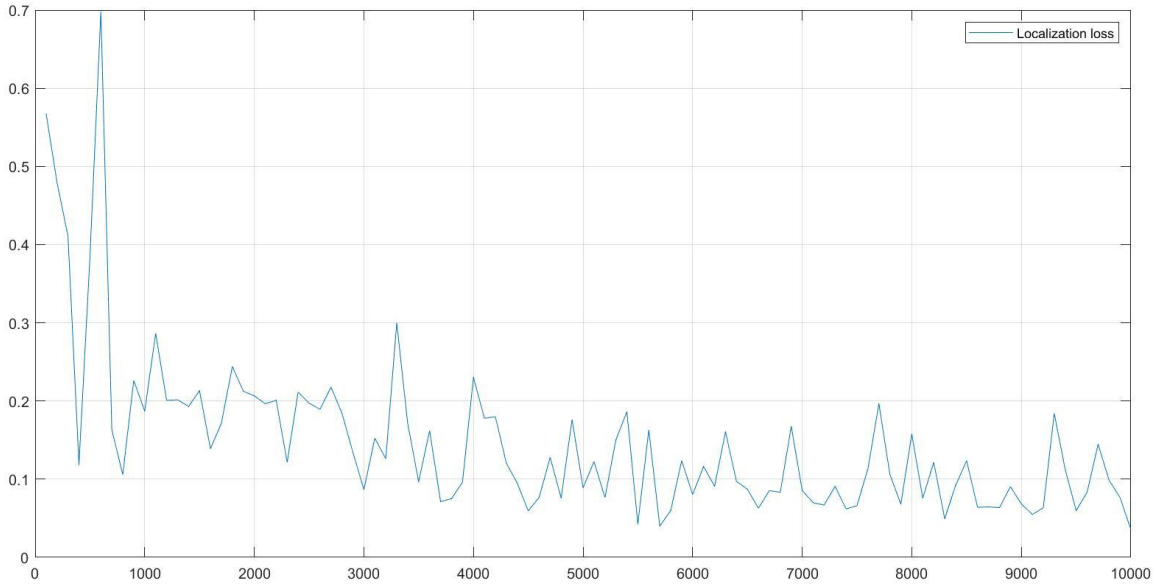
După cum se poate observa în Figura 39, discutăm despre un model stabil, a cărui rată de învățare nu prezintă schimbări majore pe parcursul procesului de antrenare.



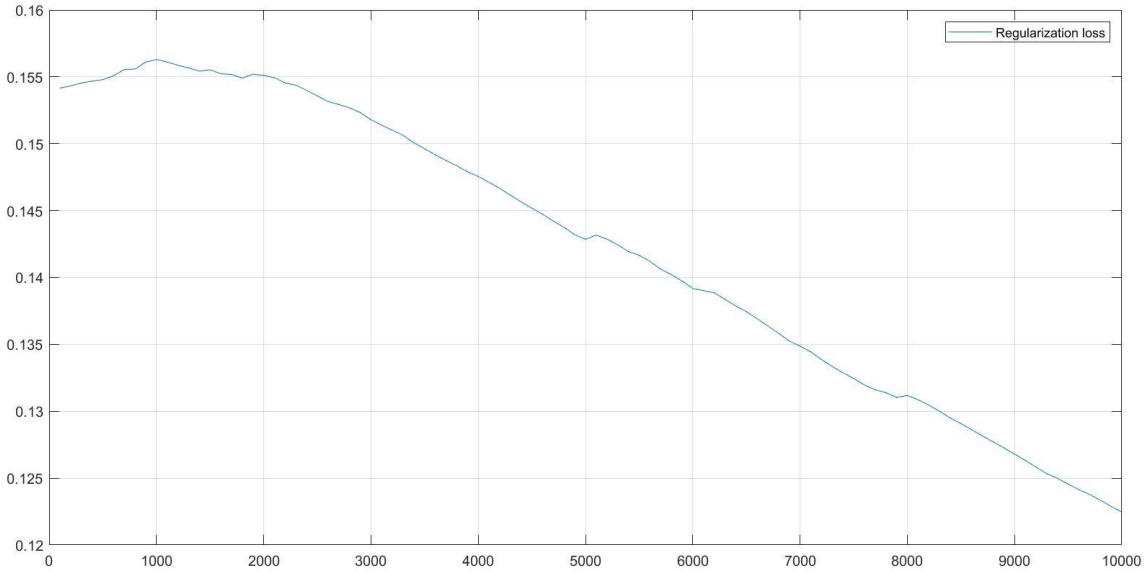
Figură 44 - Pierderea de clasificare în funcție de pasul de antrenare

De asemenea, stabilizarea modelului se observă și pe graficul pierderii de clasificare (Figura 40), această pierdere scăzând gradual odată cu progresul pasului de antrenare.

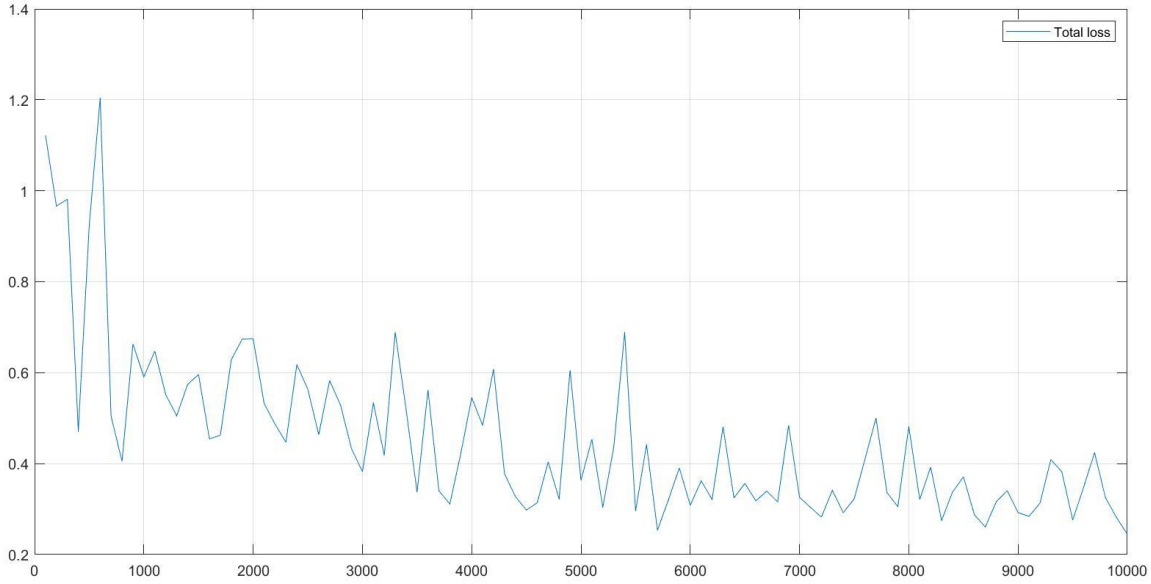
Figurile 41, 42 și 43 prezintă aceeași dependință invers proporțională a pierderilor de localizare, regularizare și a pierderilor totale în funcție de progresul procesului de învățare.



Figură 45 - Pierderea de localizare în funcție de pasul de antrenare



Figură 46 – Pierderea de regularizare în funcție de pasul de antrenare



Figură 47 – Pierderea totală în funcție de pasul de antrenare

7. CONCLUZII

7.1 UTILITATE

În această lucrare, folosind concepte moderne utilizate în domeniul Sistemelor Inteligente de Transport, am dezvoltat un sistem care detectează plăcuțe de înmatriculare de la o resursă de tip imagine sau din fluxul video preluat în timp real de către o cameră web conectată unui Raspberry Pi.

Aceste detecții sunt procesate folosind un script Python, în vederea returnării numărului de înmatriculare și a determinării în timp real a cazului în care vehiculul atașat detecției are un comportament suspicios.

Considerând resursele limitate (hardware, set de date, restricționări GDPR), sistemul returnează rezultatele așteptate, mulțumitoare, lăsând loc pentru îmbunătățiri viitoare.

De asemenea, proiectul deschide calea către o multitudine de alte soluții inteligente asemănătoare, ce pot fi implementate adaptând cazul actual.

7.2 DEZVOLTĂRI ULTERIOARE

„If you want to make an apple pie from scratch, you must first create the universe”
„Dacă vrei să faci o plăcintă cu mere de la zero, trebuie mai întâi să creezi universul”
- Carl Sagan

Pentru a atinge nivelul unui sistem absolut complet, este imperioasă dezvoltarea tuturor componentelor acestuia, odată cu avansul rapid al tehnologiilor actuale. În ritmul accelerat al erei noastre, finalul unui proiect nu poate fi atins niciodată, dezvoltări noi fiind create în mod continuu.

Dezvoltarea și îmbunătățirea nu doar a modelului de detecție automată a urmăritorilor din trafic, ci și a domeniului recunoașterii automate a numerelor de înmatriculare este impetuos necesară.

Natural, primul pas în cadrul unor dezvoltări ulterioare ar fi îmbunătățirea acurateței detecțiilor și a recunoașterii caracterelor. Tehnologiile vor continua să avanseze, iar metodele folosite în acest proiect vor putea fi actualizate, conducând la îmbunătățirea modelului. Performanța detecțiilor ar fi schimbată semnificativ în urma utilizării unui set de date extins și divers, dar și prin actualizarea soluțiilor hardware și software folosite.

Devoltată în continuare ar putea fi și interfața grafică, ce ar putea fi mai user-friendly, mai atractivă și mai completă, înglobând noi facilități. Integrarea unor API-uri în cadrul proiectului ar putea adăuga o multitudine de facilități cheie.

O opțiune folositoare în rularea proiectului este constituită de reținerea locației unei detecții. Zonele în care un număr semnificativ de detecții a urmăritorilor au fost prelucrate

ar putea fi evitate, aplicația atenționându-te și oferindu-ți căi ocolitoare. De asemenea, aplicația ar putea să ofere calea către cea mai apropiată locație sigură (determinată de utilizator sau o instituție publică precum un spital sau o secție de poliție) în momentul în care utilizatorul este urmărit în trafic.

Adițional, sistemul ar putea fi folosit de către grupuri extinse de oameni, ce ar putea vizualiza în timp real locațiile și statusul celor din grupul acestora. În momentul în care utilizatorul este în contact cu o mașină considerată periculoasă, restul utilizatorilor din grup ar putea fi și ei atenționați.

Aceste atenționări și avertizări ar putea fi trimise și via sms sau via interfață grafică, iar sistemul ar putea chiar aștepta un răspuns de la utilizator. De asemenea, utilizatorul ar putea oferi și alte input-uri, precum mașini de ignorat sau mașini asupra cărora să fie atenționat instant în momentul în care apar în aria vizuală a camerei web.

Pe lângă actualizarea și îmbunătățirea modelului, dezvoltări viitoare ale sistemului pot include folosirea unor interfețe pentru interacțiunea dintre utilizator și program. Este necesar ca aplicația să poată fi folosită din cadrul a diferite sisteme de operare, precum iOS[44], macOS[45], Android[46], Linux[47] sau Microsoft Windows[48].

De asemenea, proiectul poate constitui baza pentru alte aplicații, precum sisteme de monitorizarea a traficului, a semafoarelor dintr-o localitate, monitorizare a unor parcări inteligente, ș.a.

[44] iOS. Disponibil: https://www.apple.com/ios/ios-15/ [Accesat: 08.06.2022]
[45] macOS. Disponibil: https://www.apple.com/ro/macos/what-is/ [Accesat: 08.06.2022]
[46] Android. Disponibil: https://www.android.com/ [Accesat: 08.06.2022]
[47] Linux. Disponibil: https://www.linux.org/ [Accesat: 08.06.2022]
[48] Windows. Disponibil: https://www.microsoft.com/ro-ro/windows [Accesat: 08.06.2022]

Anexă – Cod

Codul sursă dezvoltat în cadrul proiectului, alături de documentație, seturi de date și alte documente este disponibil în repository-ul public de la următoarea adresă:

https://github.com/nasteeex/Licenta

Adițional, repository-ul poate fi accesat și prin scanarea următorului cod QR:

Referințe bibliografice

[1] Wikipedia, „automatic number-plate recognition,” 1 August 2021. [Interactiv]. Available: https://en.wikipedia.org/wiki/Automatic_number-plate_recognition. [Accesat 4 June 2022].

[2] S. Du, M. Ibrahim, M. Shehata și W. Badawy, „Shan Du; IntelliView Technol., Inc., Calgary, AB, Canada; Lakshman, M.; Shehata, M.; Badawy, Wael; Automatic License Plate Recognition (ALPR): A State-of-the-Art Review,” *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 23, nr. 2, pp. 311-325, 11 March 2017.

[3] CCTV Information, „An Introduction to ANPR,” [Interactiv]. Available: https://www.cctv-information.co.uk/article/an-introduction-to-anpr/. [Accesat 4 June 2022].

[4] ANPR International, „History of ANPR,” [Interactiv]. Available: . http://www.anpr-international.com/history-of-anpr/. [Accesat 4 June 2022].

[5] BBC, „CCTV network tracks 'getaway car',” 21 November 2005. [Interactiv]. Available: http://news.bbc.co.uk/2/hi/uk_news/england/bradford/4455918.stm. [Accesat 4 June 2022].

[6] H. M., „https://www.illawarramercury.com.au/story/113726/mass-surveillance-system-nicks-drivers/,” Illawarra Mercury, 6 June 2012. [Interactiv]. Available: https://www.illawarramercury.com.au/story/113726/mass-surveillance-system-nicks-drivers/. [Accesat 4 June 2022].

[7] Wayback machine, „web.archive.org,” [Interactiv]. Available: https://web.archive.org/web/20080409035619/http://www.extremecctv.com/pdf/productnews/CN040422-Extreme-CCTV-Announces-Contract-for-Stockholm-Traffic-Cameras.pdf . [Accesat 4 June 2022].

[8] Dutch Attorney General, „www.om.nl,” [Interactiv]. Available: https://www.om.nl/onderwerpen/verkeer/veelgestelde_vragen/trajectcontrole/. [Accesat 4 June 2022].

[9] „License Plate Recognition, Tribal Casinos, and banned Persons,” [Interactiv]. Available: https://www.indiangaming.com/istore/Feb11_Wanser.pdf. [Accesat 4 June 2022].

[10] www.thenewspaper.com, „UK Billvoards Equipped with License Plate Spy Cameras,” 25 September 2009. [Interactiv]. Available: http://www.thenewspaper.com/news/29/2907.asp. [Accesat 4 June 2022].

[11] G. J., „PREPARED FOR THE CALIFORNIA AIR RESOURCES BOARD AND THE CALIFORNIA ENVRONMENTAL PROTECTION AGENCY," *MICROSCALE EMISSIONS MODELING SYSTEM,* pp. 96-316, 2002.

[12] DIGI24, „Aproape 6.000 de copii au dispărut de acasă în ultimul an. 106 nu au fost încă găsiți," 26 May 2021. [Interactiv]. Available: https://www.digi24.ro/stiri/actualitate/social/aproape-6-000-de-copii-au-disparut-de-acasa-in-ultimul-an-106-nu-au-fost-inca-gasiti-1541827. [Accesat 5 June 2022].

[13] Ziare.com, „Statistici ingrijoratoare: In Romania, peste 5.000 de copii au disparut de acasa in ultimul an," 25 May 2019. [Interactiv]. Available: https://ziare.com/social/romani/statistici-ingrijoratoare-in-romania-peste-5-000-de-copii-au-disparut-de-acasa-in-ultimul-an-1562918. [Accesat 5 June 2022].

[14] HotNews, „Bătaie în trafic în Capitală. A fost lovit și un poliţist în civil care intervenea pentru a aplana conflictul," 1 June 2022. [Interactiv]. Available: https://www.hotnews.ro/stiri-esential-25593099-bataie-trafic-capitala-fost-lovit-politist-civil-care-intervenea-pentru-aplana-conflictul.htm. [Accesat 5 June 2022].

[15] A. Pavel și A. Popescu, „VIDEO Bătaie în trafic pe Valea Prahovei. Zece persoane s-au încăierat pe DN1, în zona Comarnic/ Printre protagoniști, o femeie cu un copil în brațe," 3 January 2022. [Interactiv]. Available: https://www.g4media.ro/video-bataie-in-trafic-pe-valea-prahovei-zece-persoane-s-au-incaierat-pe-dn1-in-zona-comarnic-printre-protagonisti-o-femeie-cu-un-copil-in-brate.html. [Accesat 5 June 2022].

[16] O. Oanta, „VIDEO. Bătaie în traficul din București. Un curier pe bicicletă a ajuns pe capota unei mașini," 6 January 2022. [Interactiv]. Available: https://stirileprotv.ro/stiri/socant/bataie-in-traficul-din-bucuresti-un-curier-pe-bicicleta-a-ajuns-dus-pe-capota-unei-masini.html. [Accesat 5 June 2022].

[17] G. Marina, „Un fenomen periculos ia amploare. Sute de șoferi circulă fără permis în București, iar numărul lor crește," 16 March 2021. [Interactiv]. Available: https://www.digi24.ro/stiri/actualitate/evenimente/un-fenomen-periculos-ia-amploare-sute-de-soferi-circula-fara-permis-in-bucuresti-iar-numarul-lor-creste-1468984. [Accesat 5 June 2022].

[18] I. Sirbu, „Ultima nebunie pe șosele din România: înmulțirea cazurilor de șoferi fără permis. „Înainte era văzută ca o abatere foarte gravă, azi pare ceva semi-nevinovat"," 2 May 2022. [Interactiv]. Available: https://www.fanatik.ro/ultima-nebunie-pe-sosele-din-romania-inmultirea-cazurilor-de-soferi-fara-permis-inainte-era-vazuta-ca-o-abatere-foarte-grava-azi-pare-ceva-semi-nevinovat-19968005. [Accesat 5 June 2022].

[19] C. Laiu, „Violenţa domestică în România | De la teamă și rușine la speranţa că va fi bine," 26 November 2021. [Interactiv]. Available: https://semneletimpului.ro/social/familie/violenta-domestica-in-romania-de-la-teama-si-rusine-la-speranta-ca-va-fi-bine.html. [Accesat 5 June 2022].

[20] S. Dojan, „Drama femeilor abuzate și eșecul statului român în a le proteja pe timp de pandemie,” 24 October 2021. [Interactiv]. Available: https://romania.europalibera.org/a/drama-femei-abuz-domestic/31521522.html. [Accesat 5 June 2022].

[21] Digi24, „40% din ordinele de protecție sunt încălcate. În fiecare oră, o femeie sună la 112 pentru a-și reclama partenerul violent,” 27 April 2021. [Interactiv]. Available: https://www.digi24.ro/stiri/actualitate/evenimente/40-din-ordinele-de-protectie-sunt-incalcate-in-fiecare-ora-o-femeie-suna-la-112-pentru-a-si-reclama-partenerii-violenti-1509845. [Accesat 5 June 2022].

[22] N. McCarthy, „Infographic: The EU's Hotspots For People Trafficking,” 2 June 2016. [Interactiv]. Available: https://www.statista.com/chart/4947/the-eus-hotspots-for-people-trafficking/. [Accesat 5 June 2022].

[23] Asociația Națională a Evaluatorilor de Risc la Securitatea Fizică din România, „Coeficienti de criminalitate,” 2020. [Interactiv]. Available: https://www.anersf.ro/coeficienti-de-criminalitate/. [Accesat 5 June 2022].

[24] Poliția Română, „Poliția Română,” 2021. [Interactiv]. Available: https://politiaromana.ro/ro/auto-furate. [Accesat 5 June 2022].

[25] A. Radu, „INFOGRAFIC: Cate masini au fost furate in 2021?,” 11 March 2022. [Interactiv]. Available: https://www.1asig.ro/INFOGRAFIC-Cate-masini-au-fost-furate-in-2021-articol-3,100-68099.htm. [Accesat 5 June 2022].

[26] A. Toma, „Peste 150 de mașini furate în străinătate au fost descoperite în 2021 la intrarea în România. Șoferii au declarat că sunt „cumpărători de bună credinţă”,” 2 January 2022. [Interactiv]. Available: https://ziare.com/politia-de-frontiera/masini-furate-gasite-la-frontiera-2021-1718154. [Accesat 5 June 2022].

[27] A. M. Turing, „Computing machinery and intelligence,” Aberdeen University Press, Aberdeen, 1950.

[28] A. Garland, Regizor, *Ex Machina.* [Film]. United States of America: Universal Studios, 2014.

[29] Z. Mahmood, K. Khan, U. Khan, S. H. Adil, S. S. A. Ali și M. Shahzad, „Towards Automatic License Plate Detection,” *Sensors,* 7 February 2022.

[30] U.S. Department of Justice, National Institute of Justice, Automated License Plate Recognition Systems: Policy and Operational Guidance for Law Enforcement, Alexandria, Virginia, 2012.

[31] American Civil Liberties Union, YOU ARE BEING TRACKED How License Plate Readers Are Being Used To Record Americans’ Movements, 2013.

[32] Plate Recognizer, „Automatic License Plate Recognition - High Accuracy ALPR,” [Interactiv]. Available: https://platerecognizer.com/ . [Accesat 4 June 2022].

[33] DevoDep, „AXIS P1445-LE-3 LPR KIT/PLATE VERIFIER KIT IN,” [Interactiv]. Available: https://devodep.ro/camere-de-supraveghere/262446-axis-p1445-le-3-lpr-kitplate-verifier-kit-in.html. [Accesat 4 June 2022].

[34] www.leonadocompany-us.com, „The Plate Hunter Mobile License Plate Camera,” [Interactiv]. Available: https://www.leonardocompany-us.com/lpr/elsag-mobile. [Accesat 4 June 2022].

[35] eBay, „Remington ELSAG ALPR License Plate Reader MPH-900 Mobile Plate Hunter Scanner,” [Interactiv]. Available: https://www.ebay.com/itm/265115650893. [Accesat 4 June 2022].

[36] T. M. Mitchell, „Machine Learning,” Mcgraw Hill, New York, 1997.

[37] J. Brownlee, „Transfer Learning in Keras with Computer Vision Models,” Machine Learning Mastery, 14 March 2019. [Interactiv]. Available: https://machinelearningmastery.com/how-to-use-transfer-learning-when-developing-convolutional-neural-network-models/. [Accesat 4 June 2022].

[38] Analytics Vidhya, „Computer Vision to Detect License Number Plate,” 30 December 2021. [Interactiv]. Available: https://www.analyticsvidhya.com/blog/2021/12/computer-vision-to-detect-license-number-plate/. [Accesat 4 June 2022].

[39] www.learnopencv.com, „Automatic License Plate Recognition using Deep Learning,” 15 March 2022. [Interactiv]. Available: https://learnopencv.com/automatic-license-plate-recognition-using-deep-learning/?ck_subscriber_id=452195442. [Accesat 4 June 2022].

[40] A. Chaudhuri, K. Mandaviya, P. Badelia și S. K. Ghosh, „Optical Character Recognition System for Different Languages with Soft Computing,” *Studies in Fuziness and Soft Computing,* pp. 1-108.

[41] AI & Machine Learning Blog, „[Tutorial] OCR in Python with Tesseract, OpenCV and Pytesseract,” 5 December 2019. [Interactiv]. Available: https://nanonets.com/blog/ocr-with-tesseract/. [Accesat 4 June 2022].

[42] Simplilearn.com, „Top Python Machine Learning Libraries,” 25 November 2020. [Interactiv]. Available: https://www.simplilearn.com/python-machine-learning-libraries-article. [Accesat 4 June 2022].

[43] TensorFlow, „TensorFlow,” 2019. [Interactiv]. Available: https://www.tensorflow.org/. [Accesat 4 June 2022].

[44] R. Vickery, „The Python Machine Learning Ecosystem,” 11 April 2022. [Interactiv]. Available: https://towardsdatascience.com/the-python-machine-learning-ecosystem-7c05be4ac48d. [Accesat 4 June 2022].

[45] P. D. R. G. K. H. B. H. S. B. T.-Y. Lin, „Feature pyramid networks for object detection,” *Proceedings of the IEEE conference on computer vision and pattern,* pp. 2117-2125, 2017.

[46] B. Y. P. F. P. A. K. N. T. D. Minaee S, „Image Segmentation Using Deep Learning: A Survey,” *IEEE Trans Pattern Anal Mach Intell.*

[47] Praveen, „License Plate Recognition using OpenCV Python,” Medium, 9 July 2020. [Interactiv]. Available: https://medium.com/programming-fever/license-plate-recognition-using-opencv-python-7611f85cdd6c. [Accesat 4 June 2022].

[48] Wikipedia, „Numerele de înmatriculare auto în România,” [Interactiv]. Available: https://ro.wikipedia.org/wiki/Numerele_de_%C3%AEnmatriculare_auto_%C3%AEn_Rom%C3%A2nia. [Accesat 13 June 2022].

[49] C. e. C. Molder, „An Automatic Licence Plate Recognition (ALPR) System,” Bucharest.

[50] www.javatpoint.com, „Number Plate Recognition using Python - Javatpoint,” [Interactiv]. Available: https://www.javatpoint.com/number-plate-recognition-using-python. [Accesat 4 June 2022].

[51] PyImageSearch, „OpenCV: Automatic License/Number Plate Recognition (ANPR) with Python,” 21 September 2020. [Interactiv]. Available: https://pyimagesearch.com/2020/09/21/opencv-automatic-license-number-plate-recognition-anpr-with-python/. [Accesat 4 June 2022].

[52] A. Rosebrock, „Intersection over Union (IoU) for object detection,” 7 November 2016. [Interactiv]. Available: https://pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/. [Accesat 12 June 2022].

[53] J. Hui, „mAP (mean Average Precision) for Object Detection,” 7 March 2018. [Interactiv]. Available: https://jonathan-hui.medium.com/map-mean-average-precision-for-object-detection-45c121a31173. [Accesat 12 June 2022].

[54] Openalpr.com, „OpenALPR - Automatic License Plate Recognition,” 2020. [Interactiv]. Available: https://www.openalpr.com/. [Accesat 4 June 2022].

[55] [Interactiv]. Available: https://www.statista.com/chart/4947/the-eus-hotspots-for-people-trafficking/.