Dezvoltarea unui instrument digital pentru cuantificarea și clasificarea entităților de trafic

Student: Ing. Roberto-Marian OPIȚEANU

Profesor coordonator : Prof. Dr. Ing. Florin Valentin RUSCĂ

Universitatea Națională de Știință și Tehnologie Politehnica București
Facultatea de Transporturi
Departamentul Transport și Trafic Urban

Rezumat

Urbanizarea accelerată și creșterea traficului rutier au generat nevoia unor soluții inteligente pentru monitorizarea și gestionarea fluxurilor de vehicule. Lucrarea de față își propune dezvoltarea unui instrument digital care utilizează rețele neuronale de tip YOLO, bazate pe învățare profundă, pentru cuantificarea și clasificarea vehiculelor în scenarii reale de trafic. În prima etapă este descris cadrul teoretic și analiza metodelor existente de detecție automată, urmată de colectarea unui set de date personalizat, etichetarea imaginilor și antrenarea modelului YOLOv8 pe cinci clase de vehicule.

Obiectivul principal al cercetării este de a dezvolta un model robust care să fie capabil să clasifice vehiculele în categorii specifice și să furnizeze un număr precis al acestora, aplicabil în taxare, analiza traficului și întreținerea drumurilor.

Modelul a fost antrenat pe un set de date personalizat, cu imagini etichetate de vehicule (vehicule personale, camioane, autobuze, tramvaie) din trafic urban. După antrenarea modelului YOLOv8 pentru 200 de iterații, se evaluează performanța acestuia utilizând valori standard (precizie, recall, mAP) și se compară rezultatele obținute cu un model YOLOv8 pre-antrenat pentru a evalua capacitatea de generalizare.

Într-o fază ulterioară a studiului, se transformă vehiculele detectate în entități de trafic etalon (VE) pe baza standardelor de trafic, pentru a oferi o estimare mai precisă a fluxului de trafic de pe o arteră. Rezultatele obținute demonstrează că modelul YOLOv8 personalizat are o performanță mai bună în detectarea vehiculelor personale, autobuzelor, camioanelor și a tramvaielor, cu un scor F1 de 86% și un mAP de peste 90% comparativ cu modelul pre-antrenat.

Performanța obținută demonstrează fezabilitatea metodei propuse și oferă o bază solidă pentru extinderea acesteia în aplicații reale.

Transformarea vehiculelor în vehicule etalon îmbunătățește semnificativ estimările fluxului de trafic, oferind un instrument util pentru aplicații în orașele inteligente. Lucrările viitoare se vor concentra pe extinderea setului de date și implementarea modelului pe dispozitive pentru monitorizarea traficului în timp real.

Lucrarea contribuie astfel la digitalizarea proceselor de monitorizare urbană prin instrumente bazate pe inteligență artificială.

Cuvinte cheie – YOLO , Detectare vehicule, Sisteme Inteligente de Transport, Vehicule Etalon

I. INTRODUCERE

Urbanizarea rapidă și creșterea numărului de vehicule sunt probleme majore cu care se confruntă multe orașe din întreaga lume. Aceste schimbări duc la congestii severe ale traficului, emisii de poluanți și un impact negativ asupra infrastructurii rutiere. În acest context, sistemele inteligente de transport (ITS) devin tot mai importante pentru gestionarea traficului și optimizarea fluxurilor de vehicule. O componentă esențială a acestor sisteme este monitorizarea și clasificarea vehiculelor, care permite autorităților să obțină informații precise despre tipurile de vehicule prezente pe drumuri și să ajusteze strategiile de gestionare a traficului.

Tradițional, metodele de clasificare a vehiculelor se bazează pe tehnici de detecție fizică, precum buclele inductive sau senzori care presupun instalarea unor echipamente pe drumuri. Aceste metode sunt, însă, limitate din punct de vedere al costurilor, preciziei și capacității de a colecta date în timp real. De asemenea, nu sunt capabile să facă distincția între tipuri de vehicule sau să proceseze volumul mare de date generate de orașele mari [1].

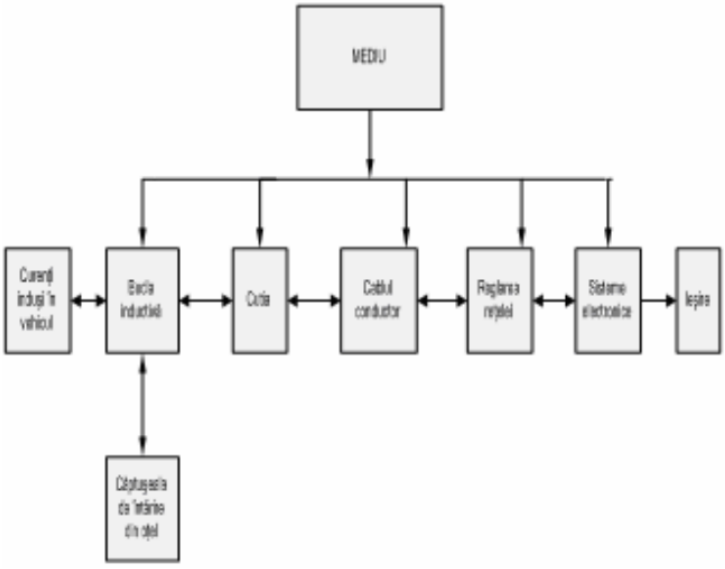

Figura 1: Modelul sistemului cu buclă inductive [2]

O soluție promițătoare pentru aceste provocări sunt modelele de învățare profundă (deep learning), în special rețelele neuronale convoluționale (CNN), care au demonstrat performanțe excelente în sarcini de recunoaștere a obiectelor. Un astfel de model, YOLO (You Only Look Once), a fost introdus pentru prima dată în 2015 și a revoluționat procesul de detecție a obiectelor prin integrarea localizării și clasificării într-o singură etapă, având un timp de procesare extrem de scurt [3]. Datorită acestui avantaj, YOLO a fost adoptat rapid în domeniul detecției vehiculelor, fiind aplicat în multe cazuri de analiză a traficului urban.

Un alt aspect important este transformarea clasificării vehiculelor în unități standardizate de măsurare a impactului asupra infrastructurii. Acest lucru poate fi realizat prin utilizarea entităților de trafic etalon, care sunt definite pe baza tipului de vehicul și a impactului său asupra traficului [4]. De exemplu, un camion poate fi echivalat cu 1.75 vehicule de tip autoturism, iar un autobuz cu 3 autoturisme. Astfel, prin transformarea numărului de vehicule

detectate în VE, se obține o estimare mai precisă a încărcăturii rutiere, care poate fi utilizată pentru analiza traficului și pentru aplicarea unor politici de taxare rutieră mai eficiente.

Tabelul 1:Coeficienții de transformare a vehiculelor în entități de trafic etalon [5]

Vehicle Type	Passenger Car Units			
	Rural Roads	Urban Roads	Roundabouts	Traffic Signals
Cars & light vans	1.0	1.0	1.0	1.0
Heavy Vehicles	3.0	1.75	2.8	1.75
Buses & Coaches	3.0	3.0	2.8	2.25
Motorcycles	1.0	0.75	0.75	0.33
Pedal cycles	0.5	0.33	0.5	0.2

În această lucrare, se propune o metodă bazată pe modelul YOLOv8 pentru clasificarea și cuantificarea vehiculelor în trafic, aplicată unui set de date urban. În continuare, sunt prezentate principiile și tehnicile relevante în acest domeniu, precum și cercetările anterioare care au contribuit la dezvoltarea și aplicarea acestor tehnologii.

II. TRECEREA ÎN REVISTĂ A LITERATURII

Detectarea obiectelor a înregistrat progrese semnificative odată cu apariția învățării profunde, în special în aplicații precum conducerea autonomă, monitorizarea traficului și supravegherea. Metodele tradiționale se bazau pe caracteristici create manual (de exemplu, cascade Haar, HOG) combinate cu clasificatori precum SVM-uri [6]. Cu toate acestea, introducerea rețelelor neuronale convoluționale (CNN-uri) a revoluționat domeniul, permițând învățarea end-to-end a sarcinilor de detectare.

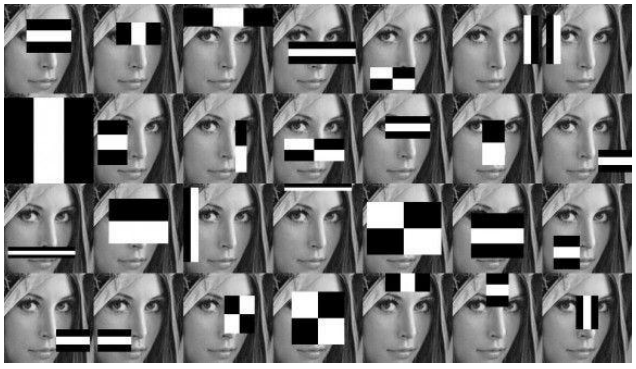

Figura 2: cascadă Haar pentru antrenarea modelelor (sursa:: https://medium.com/analytics-vidhya/haar-cascades-explained-38210e57970d)

Evoluția detectoarelor de obiecte bazate pe învățarea profundă poate fi împărțită în două abordări: cele cu două etape și cele cu o singură etapă. Detectoarele cu două etape, precum R-CNN Fast R-CNN și Faster R-CNN [7] generează mai întâi propuneri de regiuni și le clasifică

ulterior, obținând o precizie ridicată, dar la costul eficienței computaționale. În timp ce aceste metode au stabilit repere în precizie, complexitatea lor le face mai puțin potrivite pentru aplicații în timp real, precum urmărirea vehiculelor.

În schimb, detectoarele cu o singură etapă prioritizează viteza, eliminând pasul de propunere a regiunii. Modelele din familia YOLO (You Only Look Once) [3] sunt remarcabile pentru performanța lor în timp real. YOLO tratează detectarea obiectelor ca pe o problemă de regresie, prezicând casete de delimitare și probabilități de clasificare într-o singură trecere prin rețea. Îmbunătățirile ulterioare, cum ar fi YOLOv3 și YOLOv4 , au introdus îmbunătățiri precum rețelele piramidale de caracteristici (FPN), optimizarea boxelor de ancorare și funcții de pierdere modificate (de exemplu, pierderea CIoU). Cele mai recente iterații, YOLOv7 și YOLOv8, optimizează în continuare compromisurile între viteză și precizie prin rafinamente arhitecturale și strategii de instruire.

Studiile comparative evidențiază avantajele YOLO în detectarea vehiculelor. De exemplu, Liu et al. (2020) au demonstrat că YOLOv4 a depășit Faster R-CNN în ceea ce privește viteza de procesare, menținând în același timp o precizie competitivă pe setul de date UA-DETRAC. În mod similar, Zhang et al. (2021) au evaluat YOLOv5 pe imagini de trafic capturate cu drone, observând robustețea acestuia în detectarea vehiculelor mici și neclare. Aceste studii subliniază compatibilitatea YOLO pentru aplicațiile din lumea reală, unde latența este esențială.[8]

Seturile de date cheie pentru cercetarea în detectarea vehiculelor includ:

- **KITTI**: Imagini de înaltă rezoluție din scenarii de conducere autonomă. [9]
- **UA-DETRAC**: Set de date de mari dimensiuni cu vehicule annotate în trafic urban. [10]
- **COCO**: Un set de date general, cu categorii diverse de obiecte, inclusiv mașini. [11]

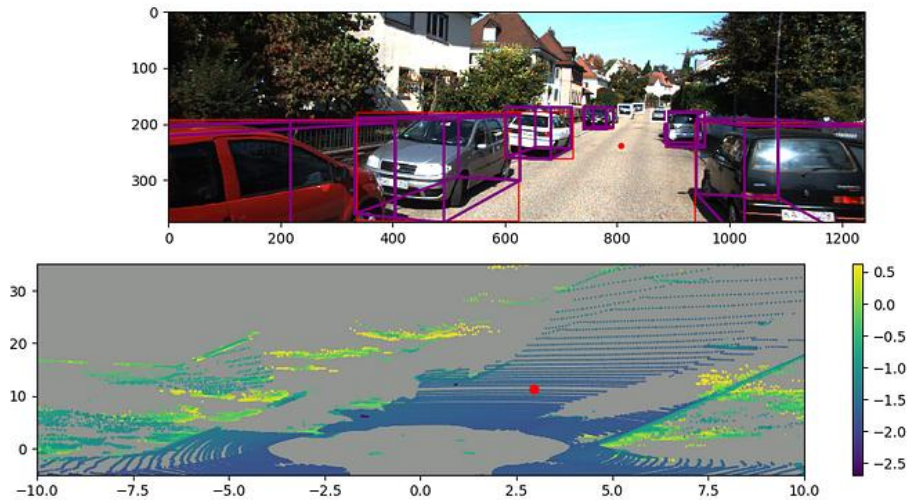

Figura 3: Set de date KITTI [9]

Măsurile de evaluare, cum ar fi Precizia Medie (mAP), cadre pe secundă (FPS) și Intersecția peste Uniune (IoU), sunt utilizate frecvent pentru a evalua performanța modelului. Lucrări recente au explorat, de asemenea, variante YOLO mai ușoare (de exemplu, YOLO-Lite,

NanoDet) pentru implementarea pe dispozitive edge, abordând constrângerile computaționale din sistemele incorporate.
În ciuda punctelor forte ale YOLO, rămân provocări în gestionarea occluziunii extreme, condițiilor meteorologice nefavorabile și adaptarea domeniului între diferite configurații ale camerelor. Direcțiile viitoare de cercetare ar putea să se concentreze pe arhitecturi hibride (de exemplu, combinarea transformatoarelor vizuale cu CNN-uri) și învățarea auto-supervizată pentru a reduce dependența de etichete.

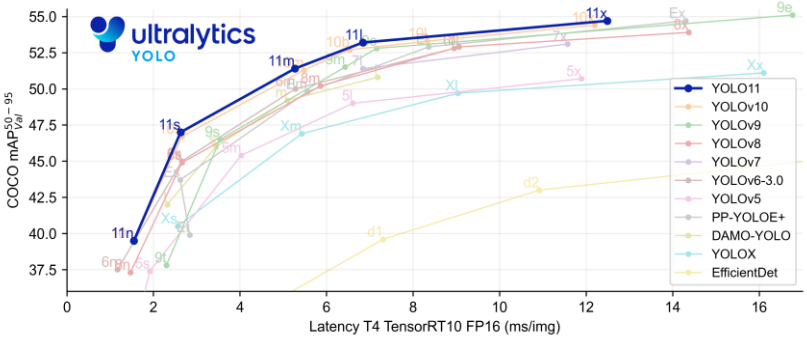

Figura 4: Graficul mediei de precizie între modelele de detectare(sursa: https://docs.ultralytics.com/models/)

III. METODOLOGIE

Setul de date

Pentru antrenarea și validarea modelului este creat un set de date personalizat, format din imagini preluate din traficul din București cu ajutorul unui program realizat în Python care împarte un video în imagini. Acestea sunt capturate din diverse unghiuri și în condiții variabile de iluminare și congestie. Setul este compus din 2489 de imagini, împărțite în 5 categorii de vehicule etichetate (car,bus,truck,bike,tram), cu formatul de etichetare YOLO .

Setul de imagini este apoi împărțit în 70% pentru antrenare, 20% pentru validare și 10% pentru testare.

Etichetarea este realizată cu ajutorul instrumentului de lucru RoboFlow, este realizat manual pentru fiecare imagine și pentru verificarea calității etichetelor se efectuează o inspecție vizuală aleatorie pentru a asigura coerența și acuratețea clasificărilor.

Figura 5: Etichetarea vehiculelor (sursa: prelucrare autor Roboflow)

Antrenarea modelului

Antrenarea modelului YOLOv8 constă în optimizarea unei funcții obiectiv care combină mai multe componente precum: eroarea de localizare (bounding box), eroarea de clasificare și penalizarea pentru detectări incorecte (false positive/negative). Pentru o imagine de intrare , unde H și W sunt înălțimea și lățimea imaginii și 3 reprezintă cele trei canale de culoare (RGB), modelul produce o ieșire care conține pentru fiecare celulă următoarele:

- Coordonatele casetei de detecție (x,y,w,h)

  unde:

- x și y sunt coordonatele casetei de detecție;
- w și h reprezintă lățimea și înălțimea predicției pentru obiectul detectat.

- Probabilitatea ca celula să conțină un obiect p_0, care indică în ce măsură modelul crede că în acea celulă există un obiect (spre deosebire de fundal)

- Distribuția de probabilitate pentru clasele de obiecte

$$p(c_i), pentru\ c_i \in \{1,2, \dots C\}$$

  unde:

- C este numărul total de clase
- $p(c_i)$ probabilitatea ca obiectul detectat să aparțină clasei c_i

Se definește o funcție obiectiv (funcție de pierdere) care cuantifică diferența dintre predicțiile rețelei și valorile reale. Această funcție are rolul de a ghida procesul de învățare prin optimizarea parametrilor rețelei neuronale.

Funcția este compusă din trei elemente principale:

$$L = \lambda_{\text{box}} \cdot L_{\text{box}} + \lambda_{\text{cls}} \cdot L_{\text{cls}} + \lambda_{\text{obj}} \cdot L_{\text{obj}} \quad (1)$$

unde:

L_{box} - pierderea pentru localizarea obiectului,exprimata prin pierderea CIoU (Complete Intersection over Unuion

L_{cls} – pierderea pentru clasificarea obiectului (Binary Cross Entropy)

L_{obj} – pierderea pentru prezența sau absența obiectului în celulă (detecție obiect vs fundal)

λ_{box}, λ_{cls}, λ_{obj} – coeficienți de ponderare (hiperparametri) care controlează influența fiecărei componente

Pentru îmbunătățirea preciziei de localizare YOLOv8 utilizează funcția CIoU (Complete IoU) care ia în considerare nu doar suprapunerea, ci și distanța dintre centre și proporțiile casetelor. Formula pierderii CIoU este:

$$L_{\text{box}} = 1 - \text{CIoU} = 1 - \left(\text{IoU} - \frac{\rho^2(b, b^{gt})}{c^2} - \alpha v \right) \quad (2)$$

unde:

$\rho^2(b, b^{gt})$ – pătratul distanței euclidiene dintre centrul casetei prezise b și caseta reală b_{gt}

c^2 – pătratul diagonalei celui mai mic dreptunghi care le conține pe ambele

v – raport lățime/înălțime (măsoara diferența de aspect)

α – factor de echilibrare între termeni

Pentru clasificarea obiectului (pe baza vectorului de clase), se utilizează Binary Cross Entropy (BCE)

$$L_{\text{cls}} = -\sum_{i=1}^{C} [y_i \cdot \log(\hat{y_i}) + (1 - y_i) \cdot \log(1 - \hat{y_i})] \quad (3)$$

unde:

y_i – eticheta reală pentru clasa i

$\widehat{y_i}$ – probabilitatea prezisă de rețea pentru clasa i

C – numărul total de clase

Pentru a decide dacă într-o celulă se află un obiect sau nu, se utilizează tot **Binary Cross Entropy**, dar aplicat scalar pentru probabilitatea obiectului

$$L_{\text{obj}} = -y_{\text{obj}} \cdot \log(\widehat{p_o}) + \left(1 - y_{\text{obj}}\right) \cdot \log(1 - \widehat{p_o}) \quad (4)$$

unde:

$y_{obj} \in \{0,1\}$; valoare binară care indică dacă există sau nu există obiecte în celulă

$\widehat{p_0}$ – probabilitatea prezisă că există un obiect

Eficientizarea funcției obiectiv se realizează prin ajustarea parametrilor θ ai rețelei folosind o regulă de tip gradient descendent:

$$\theta^{(t+1)} = \theta^{(t)} - \eta \cdot \nabla_\theta L\left(\theta^{(t)}\right) \quad (4)$$

IV. Rezultate

Pentru evaluarea performanței modelului antrenat, sunt analizate mai multe valori, precum pierderile (loss-urile) din timpul antrenării, precizia, recall-ul, precum și valorile mAP (mean Average Precision). În Figura 6 sunt ilustrate evoluțiile valorilor box_loss, cls_loss și dfl_loss atât pentru setul de antrenare, cât și pentru cel de validare, de-a lungul a 200 de iterații. Se observă o scădere progresivă și stabilă a tuturor pierderilor, ceea ce indică o convergență eficientă a modelului. Valorile train/box_loss și val/box_loss scad de la valori inițiale de peste 1.2 la aproximativ 0.3, iar comportamentul similar este observat și pentru celelalte tipuri de pierderi.

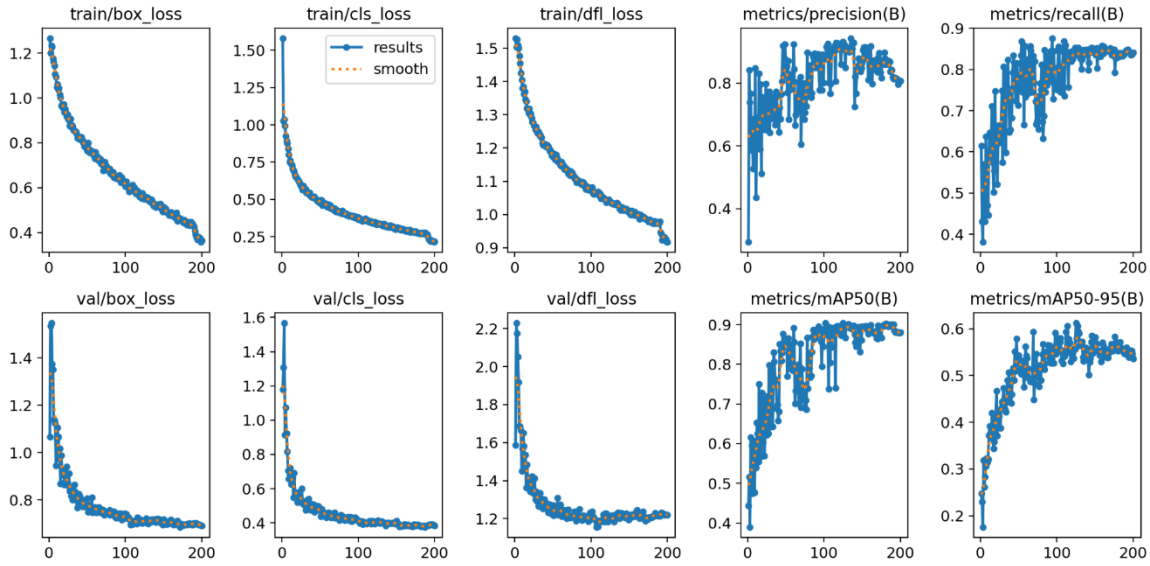

Figura 6: Reprezentarea grafică a modelului antrenat (prelucrare autor VisualCode)

În ceea ce privește performanțele modelului, metricele de evaluare precision(B) și recall(B) cresc constant pe parcursul antrenamentului, atingând valori maxime de aproximativ 0.88, respectiv 0.86. Aceste valori indică o capacitate ridicată a modelului de a detecta corect obiectele din imagini, cu o rată redusă de falsuri pozitive și falsuri negative.

Evaluarea globală a performanței este reflectată de metricele mAP. Astfel, mAP50(B) a atins o valoare maximă de peste 0.9, iar mAP50-95(B) a crescut până la aproximativ 0.6, valori ce confirmă acuratețea modelului în diverse condiții de detecție.

În Figura 7 este prezentată curba F1 în funcție de nivelul de încredere al predicțiilor, atât pentru fiecare clasă individuală, cât și pentru toate clasele cumulate. Valoarea optimă a fost atinsă la un prag de încredere de 0.738, unde F1-score-ul combinat pentru toate clasele este de 0.86. Se observă că clasele tram și truck prezintă cele mai ridicate valori ale scorului F1 (>0.9), în timp ce clasa bike are o performanță semnificativ mai scăzută, sugerând o dificultate mai mare în detectarea acesteia.

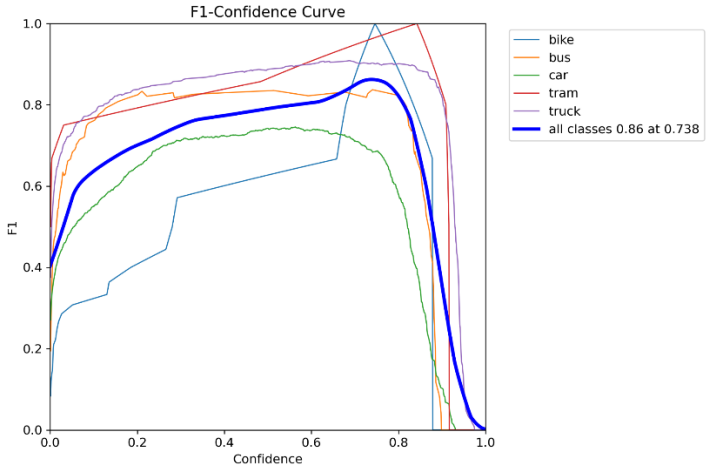

Figura 7: Curba F1 în funcție de nivelul de predicție al modelului (sursa:prelucrare autor VisualCode)

Aceste rezultate indică faptul că modelul are o performanță robustă și generalizează bine pe setul de validare, fiind astfel potrivit pentru aplicații reale de detecție a obiectelor în trafic.

Pentru a evalua performanțele aduse de antrenarea modelului pe setul de date creat se compară valorile modelului antrenat cu cele ale modelului preantrenat. Rezultatele sunt sintetizate în Tabelul 2.

Tabelul 2: Comparația valorilor modelului antrenat vs preantrenat (sursa: prelucrare autor VisualCode)

Model	mAP@0.5	mAP@0.5:0.95	Precision	Recall
YOLOv8s Preantrenat	0.1467	0.0742	0.3371	0.1274
YOLOv8s Personalizat	0.9038	0.6134	0.9322	0.8351

Se poate observa că mAP@0.5 a crescut de la 14.67% la 90.38%, iar mAP@0.5:0.95 de la 7.42% la 61.34%, evidențiind o adaptare excelentă la domeniul specific (detecția în trafic). Precision și Recall au crescut semnificativ (de la sub 34% și 13% la peste 93%, respectiv 83%), ceea ce indică o reducere drastică a falsurilor pozitive/negative.

Diferența majoră se datorează antrenării pe un set de date specializat, care capturează particularități ale obiectelor într-un scenariu de trafic real.(unghiuri specifice,vehicule rare). Scorurile scăzute ale modelului preantrenat se datorează supraadaptării la seturi de date generice, unde modelul a învățat să recunoască modele specifice, fără a generaliza bine la particularitățile traficului urban.

De asemenea, se utilizează o serie de augumentări (rotații,blur,schimbări de lumină) pentru a permite modelului să învețe caracteristici robuste.

În Figura 9 este prezentată interfața aplicației, împărțită în 2 secțiuni – analiza cantitativă și analiza video utilizată pentru coerența și gradul de încredere al programului. Aplicația permite reprezentarea grafică a zonelor (benzi, sensuri) unde vehiculele sunt contorizate. Din punct de vedere al analizei cantitative am ales să folosesc 3 categorii și anume vehiculele contorizate pe clase, vehiculele contorizate pe sensuri (zone) și fluxul de trafic exprimat în vehicule/minut.

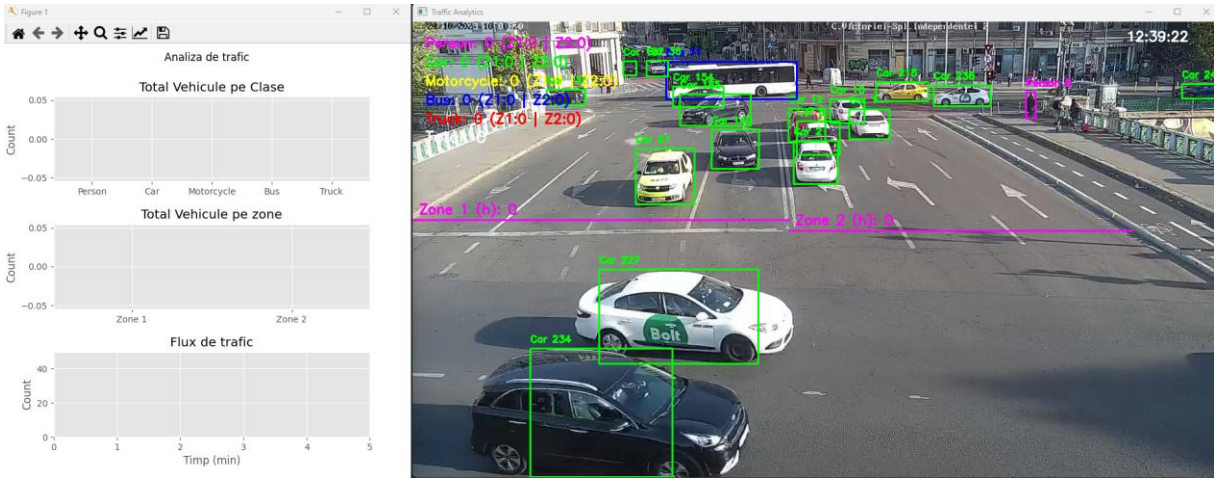

Figura 8: Interfața instrumentului digital (sursa: prelucrare autor VisualCode)

V. LIMITĂRI ȘI DIRECȚII DE CERCETARE VIITOARE

Deși modelul antrenat a demonstrat performanțe remarcabile în detecția obiectelor din trafic, analiza rezultatelor trebuie contextualizată în raport cu anumite limitări metodologice și tehnice care pot influența generalizarea soluției propuse în scenarii reale.

Limitări hardware și constrângeri computaționale

Antrenamentele au fost realizate pe o configurație hardware ce include un procesor AMD Ryzen 5 7600 (6 nuclee/12 fire de execuție), 32 GB memorie RAM DDR5 și o placă grafică AMD RX 6600 cu 8 GB memorie video dedicată. Deși această infrastructură este adecvată pentru experimente de complexitate medie, a impus restricții semnificative

asupra dimensiunii batch-urilor utilizate în procesul de antrenare, numărului total de iterații rulate și alegerea arhitecturii de rețea (optându-se pentru YOLOv8s în detrimentul variantelor mai complexe, precum YOLOv8l sau YOLOv8x).

Limitări ale setului de date

Setul de date utilizat prezintă o diversitate redusă din punct de vedere al condițiilor de mediu și conține un dezechilibru semnificativ între clase. Acest aspect este reflectat în performanțele inferioare obținute pentru clasa „bicicletă", comparativ cu alte clase precum „tramvai" sau „autobuz". Dezechilibrul între clase este cunoscut pentru impactul său negativ asupra rețelelor de detecție [12]. Mai mult, setul nu acoperă suficient scenarii extreme, cum ar fi condiții meteorologice nefavorabile, iluminare redusă sau unghiuri de captură atipice — factori esențiali în aplicațiile reale de tip vehicul autonom sau supraveghere urbană.

Lipsa evaluării pe dispozitive periferice inteligente

Un alt aspect important îl constituie absența testării modelului pe sisteme de procesare la marginea rețelei, unde resursele de procesare sunt limitate. Nu a fost analizată performanța modelului în timp real (latență de inferență), consumul energetic în scenarii de operare continuă, adaptabilitatea arhitecturii la constrângeri de memorie și procesare specifică acestor platforme [13].

Măsuri neimplementate

Din cauza limitărilor menționate anterior, nu au fost investigate tehnici avansate de optimizare care ar fi putut îmbunătăți suplimentar eficiența modelului, precum augmentări avansate ale datelor (mixup, mosaic, cutout)[14], tehnici de compresie a modelelor (e.g., pruning, cuantizare post-antrenare) [15] și transferul de cunoștințe între modele de dimensiuni diferite.

Recomandări pentru cercetări viitoare

Pentru a depăși aceste limitări și a crește aplicabilitatea practică a soluției dezvoltate, studiile viitoare pot avea în vedere:

- utilizarea infrastructurii cloud pentru antrenamente pe scară largă, cu modele de complexitate crescută [16];

- extinderea și echilibrarea setului de date, incluzând scenarii variate din punct de vedere geografic, climatic și temporal ;

- aplicarea unor metode moderne de optimizare și compresie, cu scopul de a permite rularea eficientă pe platforme edge;

- validarea în condiții operaționale reale, inclusiv în aplicații de mobilitate urbană, monitorizare video inteligentă sau sisteme autonome [17].

Implementarea acestor direcții ar putea contribui semnificativ la creșterea robusteții, generalizării și eficienței modelului într-un cadru aplicativ realist.

Referințe

[1] Wang, L., & Zhang, Y., "Economic and Environmental Costs of Urban Traffic Congestion," *Sustainable Cities and Society*, vol. 89, p. 104328, 2023.
[2] M. C. Surgiu, I. M. Moise and E. A. Stanciu, "Studiu privind metode optime de detecție a vehiculelor în București," AGIR, 2012.
[3] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. În Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 779-788.
[4] Fang, W., & Chien, S., "Optimizing Commercial Vehicle Traffic in Urban Areas," *Transportation Research Part D: Transport and Environment*, vol. 45, no. 3, pp. 112–125, 2022.
[5] Atoo, A. and Adeke, P., "Level of Service (LOS) of Freeway Segments within Makurdi Metropolis," 2018.
[6] N. Dalal and B. Triggs, “Histograms of Oriented Gradients for Human Detection,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), San Diego, CA, USA, 2005, pp. 886–893.
[7] S. Ren, K. He, R. Girshick and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137-1149, Jun. 2017.
[8] Zhang, Y., Li, Z., & Wang, Y. (2021). Evaluation of YOLOv5 for Vehicle Detection in Aerial Traffic Images. În Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, 2021, pp. 112-120.
[9] A. Geiger, P. Lenz and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 3354–3361.
[10] Wen, Z., Zhang, W., & Li, Z. (2020). UA-DETRAC: A Large-Scale Benchmark for Traffic Video Understanding. În Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, 2020, pp. 625-632.
[11] T. Y. Lin et al., "Microsoft COCO: Common objects in context," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 740–755.
[12] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 779–788.
[13] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," *arXiv preprint arXiv:1804.02767*, 2018.
[14] A. Bochkovskiy, C. Y. Wang and H. Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," *arXiv preprint arXiv:2004.10934*, 2020.
[15] J. Jocher et al., "YOLOv8: A Cutting-edge Real-Time Object Detection Model," Ultralytics, 2023.
[16] M. Cordts et al., "The Cityscapes Dataset for Semantic Urban Scene Understanding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016.
[17] J. Lin et al., "Edge AI: On-demand deep learning model co-inference with device-edge synergy," *ACM SIGCOMM*, 2019.