Sveučilište u Rijeci

Fakultet informatike i digitalnih tehnologija

Diplomski studij IIS

Detekcija objekata na cesti za autonomna vozila

Seminarski rad

Kolegij: Meko računarstvo, Računalni vid

Student: Ivan Fajta, Leo Maršić

Ak. god.: 2024./2025

Rijeka, veljača 2025.

Sadržaj

1. Uvod i Motivacija

Razvoj inteligentnih transportnih sustava i autonomnih vozila posljednjih godina značajno je ubrzao napredak u računalnom vidu, posebno u području detekcije objekata na cestama. Detekcija objekata poput vozila, pješaka, biciklista i prometnih znakova, ključna je za sigurnu i učinkovitu navigaciju autonomnih sustava. Povećanje broja vozila na cestama te sve veći zahtjevi za sigurnošću i optimizacijom prometa dodatno naglašavaju važnost točnih i pouzdanih algoritama za ovaj problem.

U ovom radu fokusirali smo se na detekciju objekata na cesti. Korišteni su modeli temeljeni na konvolucijskim neuronskim mrežama (CNN), konkretno Single Shot Detector (SSD) i YOLOv8 (You Only Look Once). Ovi modeli su odabrani zbog svoje učinkovitosti u stvarnom vremenu i visoke točnosti u zadacima detekcije objekata. KITTI dataset je korišten zbog dobrih snimki cestovnog prometa.

Kako bismo dodatno poboljšali performanse modela, primijenili smo genetski algoritam za optimizaciju hiperparametara te fuzzy logiku za donošenje odluka o kvaliteti detekcije objekata. Genetski algoritam omogućava automatsko prilagođavanje parametara modela bez potrebe za ručnim podešavanjem, dok fuzzy logika uvodi dodatnu fleksibilnost u procjeni pouzdanosti detekcija.

Cilj ovog rada je istražiti utjecaj ovih metoda na performanse SSD i YOLOv8 modela, analizirati njihove prednosti i nedostatke te utvrditi koje su kombinacije najučinkovitije za detekciju objekata na cesti.

2. Pregled područja - algoritama za navedeni zadatak

Detekcija objekata na cestama predstavlja ključnu komponentu inteligentnih transportnih sustava i autonomnih vozila. Svrha detekcije je identificirati i klasificirati objekte u stvarnom vremenu, pri čemu je ključno postići visoku točnost i brzinu. Kroz povijest, razvijene su različite metode za rješavanje ovog problema, od klasičnih metoda temeljenih na ručno definiranim značajkama do suvremenih pristupa temeljenih na dubokom učenju.

Klasične metode detekcije objekata

Prije pojave dubokih neuronskih mreža, detekcija objekata temeljila se na inženjerski definiranim značajkama i algoritmima za njihovo prepoznavanje. Najpoznatije klasične metode uključuju:

HOG (Histogram of Oriented Gradients)

HOG je metoda razvijena za prepoznavanje pješaka i drugih objekata analizom gradijenata slike.

SIFT (Scale-Invariant Feature Transform)

SIFT je metoda koja prepoznaje ključne točke objekta bez obzira na promjenu skale, rotaciju ili osvjetljenje. Koristi se za prepoznavanje objekata na temelju stabilnih značajki.

Haar Cascade

Haar Cascade koristi značajke koje se temelje na integralnim slikama i trenira se pomoću algoritma koji detektira objekte kroz hijerarhijski niz jednostavnih klasifikatora. Poznata je po svojoj primjeni u detekciji lica.

Klasične metode postigle su značajan napredak u ranijim fazama računalnog vida, ali njihova učinkovitost drastično opada u složenijim i dinamičnim okruženjima poput cestovnog prometa.

Metode detekcije objekata temeljene na dubokom učenju

Pojavom konvolucijskih neuronskih mreža (CNN) otvorene su nove mogućnosti za detekciju objekata. Za razliku od klasičnih metoda, CNN automatski uči značajke iz podataka, čime se povećava preciznost i prilagodljivost modela. Razlikujemo nekoliko značajnih arhitektura:

R-CNN (Regions with CNN features)

R-CNN generira potencijalne regije na slici, a zatim koristi CNN za njihovu klasifikaciju.

Fast R-CNN

Poboljšana verzija R-CNN-a koja koristi zajedničku CNN mrežu za cijelu sliku, smanjujući vrijeme obrade.

Za ovaj rad koristili smo SSD i YOLOv8 zbog njihove učinkovitosti u real-time detekciji. Ovi modeli omogućuju brzu obradu podataka, što je ključno u prometu.

SSD (Single Shot Detector)

SSD koristi višeslojnu arhitekturu kako bi detektirao objekte na različitim razinama slike. Svaki sloj fokusira se na objekte različitih veličina, što omogućuje balansiranje između točnosti i brzine.

Prednosti: Brži od R-CNN modela i učinkovit za real-time aplikacije.

Nedostaci: Teže prepoznaje manje objekte zbog prirode višeslojne arhitekture.

YOLOv8

YOLOv8 generacija YOLO algoritama sadrži optimizacije u arhitekturi, napredne mehanizme za predviđanje bounding boxova i skalabilnost. Model koristi "anchor-free" pristup i poboljšane loss funkcije.

Prednosti: Izuzetna brzina i točnost, pogodan za složene scenarije.

Nedostaci: Kod preklapanja objekata može doći do smanjenja preciznosti.

Zaključno, razvoj detekcije objekata prošao je značajan put od klasičnih metoda do dubokih neuronskih mreža. Odabrani modeli SSD i YOLOv8 predstavljaju suvremene, učinkovite i brze pristupe detekciji objekata, pri čemu dodatna primjena genetskog algoritma i fuzzy logike može dodatno unaprijediti njihove performanse.

3. Opis baze (KITTI dataset)

Za potrebe ovog rada koristili smo KITTI dataset, jedan od najpoznatijih i najkorištenijih skupova podataka u području računalnog vida i autonomne vožnje. Ova baza podataka razvijena je na Institutu za računalnu znanost Sveučilišta u Karlsruheu u suradnji s Toyota Technological Institutom u Chicagu. Glavni cilj KITTI dataset-a je pružiti raznolik i realističan skup podataka koji simulira stvarne uvjete vožnje u urbanim, prigradskim i ruralnim prometnim okruženjima.

Namjena KITTI dataset-a:

KITTI dataset je namijenjen razvoju, testiranju i evaluaciji algoritama za:

Detekciju i klasifikaciju objekata na cesti

Praćenje objekata kroz vremenske okvire (object tracking)

Semantičku i instancijsku segmentaciju prometne scene

Procjenu dubine (depth estimation) i odometriju vozila

Planiranje putanje i lokalizaciju autonomnih sustava

Zahvaljujući svojoj sveobuhvatnosti, KITTI dataset često se koristi kao referentna točka za usporedbu performansi različitih modela računalnog vida, uključujući SSD i YOLO modele koji su korišteni u ovom radu.

Struktura i sadržaj baze podataka

KITTI dataset prikupljen je tijekom vožnje automobila kroz različite dijelove Karlsruhea i njegove okolice. Sustav za prikupljanje podataka uključivao je napredne senzore:

Kamere: Dvije stereo kamere visoke rezolucije (RGB) postavljene na prednjoj strani vozila za simulaciju ljudskog vida.

LiDAR senzor: Velodyne HDL-64E za prikupljanje 3D point cloud podataka.

GPS i IMU sustav: Za preciznu lokalizaciju i mjerenje dinamike vozila.

Mi koristimo 5200 train slika, 1481 validation slika i 800 test slika.

Podaci u KITTI dataset-u obuhvaćaju:

RGB slike – korištene za detekciju objekata u dvodimenzionalnom prostoru.

Anotacije objekata: Ručne anotacije za različite klase objekata (automobili, pješaci, biciklisti itd.), koje uključuju informacije o koordinatama bounding boxova, klasama objekata i orijentaciji.

Kalibracijski podaci: Za usklađivanje kamera i senzorskih podataka.

Opis korištenih dijelova baze

Klase objekata: Car, Van, Truck, Pedestrian, Person_sitting, Cyclist, Tram.

Anotacije objekata: koordinate bounding boxova, klase objekata i orijentacija

Izazovi i posebnosti KITTI dataset-a

Različite veličine objekata: Pješaci i biciklisti često su manji u odnosu na vozila, što predstavlja izazov za modele poput SSD-a.

Preklapanje objekata: Na urbanim raskrižjima često dolazi do preklapanja vozila, što otežava ispravno pozicioniranje bounding boxova.

KITTI dataset predstavlja izuzetno kvalitetan skup podataka za razvoj i testiranje algoritama za detekciju objekata na cesti. Njegova raznolikost i kvaliteta podataka omogućili su nam detaljnu evaluaciju performansi SSD i YOLOv8 modela, kao i analizu učinka genetskog algoritma i fuzzy logike na njihovu učinkovitost.

4. Modeli – Arhitektura

Programski alati:

- Python 3.9
- Tensorflow 2.10.0 -gpu

YOLO

Koristimo yolov8n.pt koji je pred treniran model YOLOv8 koji sadrži:

- Arhitekturu modela – Definiciju slojeva neuronske mreže.
- Težine modela – Naučene parametre
- Konfiguracijske postavke

Predprocesiranje

Skripta konvertira KITTI oznake u YOLO format prilagođavanjem koordinata bounding boxova.

Povezuje nazive klasa iz KITTI formata s odgovarajućim ID-evima u YOLO formatu.

- Učitava bounding box u KITTI formatu (x1, y1, x2, y2).
- Konvertira ga u YOLO format (x_center, y_center, width, height)
- Sprema rezultate u odgovarajući YOLO .txt datoteku

.yaml datoteka se koristi kao pokazivač za YOLO model

Glavna Skripa

1. Postavljanje okoline

   Provjerava dostupnost GPU-a i postavlja TensorFlow tako da optimizira memoriju ako je moguće.

   Sprječava multiprocessing grešku na Windows sustavu.

2. Treniranje YOLO modela

   Koristi predtrenirani YOLOv8 Nano model za početak treninga.

   Nakon treninga, učitava se najbolji (best.pt) trenirani model iz direktorija rezultata.

3. Testiranje modela

   Učitava testne slike iz direktorija i provodi detekciju objekata pomoću treniranog modela.

   Detektirane slike sprema u izlazni direktorij

4. Evaluacija modela

   Model se evaluira na testnom skupu

   Ispisuju se evaluacijske vrijednosti

Genetski Algoritam

Koritili smo genetski algoritam za optimizaciju hiperparametara prilikom treniranja YOLOv8 modela na KITTI datasetu.

Skripta za određivanje parametara:

1. Definicija hiperparametara

   Postavljeni su rasponi za batch size, veličinu slike, learning rate i optimizator.

   Generira se nasumična kombinacija hiperparametara za eksperimentiranje.

2. Evaluacija modela

Trenira YOLO model koristeći generirane hiperparametre.

Nakon treninga, provodi evaluaciju na testnom skupu i vraća mAP@50 kao mjernu vrijednost.

3. Genetski algoritam

Inicijalizira populaciju s nasumično generiranim hiperparametrima.

Evolucija kroz generacije:

Treniraju se i evaluiraju modeli.

Selektiraju se najbolji modeli (gornjih 50%).

Stvaraju se novi potomci križanjem i mutacijom hiperparametara.

Fuzzy logika

Primjenjivanje fuzzy logike za analizu pouzdanosti detekcija.

Ulazi: confidence i size

Izlaz: decision

Svaku detekciju procjenjuje fuzzy sustav i kategorizira u nisko, srednje ili visoko povjerenje.

Grupiranje detekcije po klasama i razinama povjerenja.

Rezultati se spremaju u "detection_summary.txt".

Model se evaluira na testnom skupu uz metrike:

mAP@50, mAP@50-95, preciznost, recall, F1 score, vrijeme inferencije

SSD

Počeli smo graditi model sa „ssd_mobilenet_v2_fpnlite_320x320_coco17_tpu-8“ koji je pred-trenirani model za objektno prepoznavanje.

- MobileNetV2 – lagana konvolucijska neuronska mreža
- FPNLite (Feature Pyramid Network Lite) – poboljšana struktura mreže za bolje prepoznavanje objekata različitih veličina.
- 320x320 – ulazne slike su veličine 320x320 piksela.
- COCO 2017 – treniran na COCO dataset-u (Common Objects in Context)

Predprocesiranje podataka

1. Skripta za konverziju YOLO formata anotacija u COCO JSON format.

Za svaku YOLO anotaciju:

Učitava se slika i dobivaju dimenzije

YOLO bounding box se mijenja u COCO bounding box

Kreira se COCO JSON struktura sa:

"images": informacije o slici

"annotations": anotacije objekata

"categories": lista kategorija

Sprema se u JSON datoteku.

2. Skripta koja COCO anotacije mijenja u TFRecord format koji je potreban za treniranje modela u TensorFlow Object Detection API

Učitava sliku

Ekstraktira bounding box anotacije

Konvertira ih u TFRecord format

Generira TFRecord datoteke

Kreira label_map.pbtxt koji povezuje ID klase sa njenim imenom

Struktura modela SSD:

- exporter_main_v2.py – export gotovog modela iz checkpointa
- model_main_tf2.py – skripta SSD model koja vodi treniranje ili testiranje
- kitti_u_coco.py – pretvorba KITTI formata podataka u COCO
- format_tfrecord.py – stvaranje tfrecord datoteka za funkciju SSD modela
- pipeline.config – datoteka za sve postavke i parametre SSD modela

Glavna skripta modela SSD

Postavljanje TensorFlow GPU konfiguracije

Kod detektira dostupne GPU uređaje i postavlja memory growth kako bi se izbjegli problemi sa RAM-om.

Treniranje SSD modela

Koristi model_main_tf2.py za pokretanje treninga.

Koristi pipeline.config file za podešavanje hiperparametara.

Eksportiranje modela u SavedModel format

Poslije treninga, model se eksportira u SavedModel format pomoću exporter_main_v2.py.

Ovo omogućava da se kasnije koristi za inferenciju i testiranje.

Testiranje modela na test slikama

Učitava eksportirani model pomoću tf.saved_model.load().

Izvodi detekciju objekata i crta bounding boxeve na slikama.

Čuva slike sa i bez bounding boxeva

Evaluacija modela

Evaluacija se pokreće pomoću model_main_tf2.py

Model se testira na test skupu i prikazuju se rezultati performansi.

Genetski algoritam SSD modela

Inicijalizacija populacije te se stvara početna grupa nasumičnih jedinki unutar definiranih raspona.

Genetski optimizator testira sljedeće hiperparametre unutar definiranih raspona:

- l2_regularizer_weight – Regularizacija modela
- gamma – Parametar fokalne gubitne funkcije
- alpha – Balansiranje pozitivnih/negativnih uzoraka u fokalnoj gubitnoj funkciji
- learning_rate_base – Početna stopa učenja
- warmup_learning_rate – Početna stopa učenja tijekom zagrijavanja
- warmup_steps – Broj koraka za warmup fazu
- momentum_optimizer_value – Vrijednost momentuma za optimizator

Evaluacija – Za svaku jedinku:

Generira se odgovarajući pipeline.config s hiperparametrima

Pokreće se trening modela

Nakon treninga, pokreće se evaluaciju i dohvaća mAP iz TensorBoard logova.

Selekcija – Odabire najbolje jedinke (one s najvećim mAP)

Crossover i mutacija – Kombinira parametre odabranih jedinki kako bi stvorio novu generaciju.

Proces se ponavlja kroz više generacija, poboljšavajući hiperparametre modela kroz evoluciju.

Na kraju se ispisuje optimalni skup hiperparametara koji je postigao najbolji mAP.

Fuzzy logika

Primjenjivanje fuzzy logike za analizu pouzdanosti detekcija.

Obrada detekcija

Parsira label_map.pbtxt kako bi mapirao ID klasa na njihova imena

Koristi fuzzy logiku za ocjenjivanje kvaliteta detekcije na osnovu confidenc-a

Postavlja adaptivni prag detekcije u zavisnosti od kvaliteta predikcije.

Spremanje rezultata

Sprema rezultate detekcije u detection_log.txt.

Ispisuje statistiku po klasama i confidence razinama (low, medium, high).

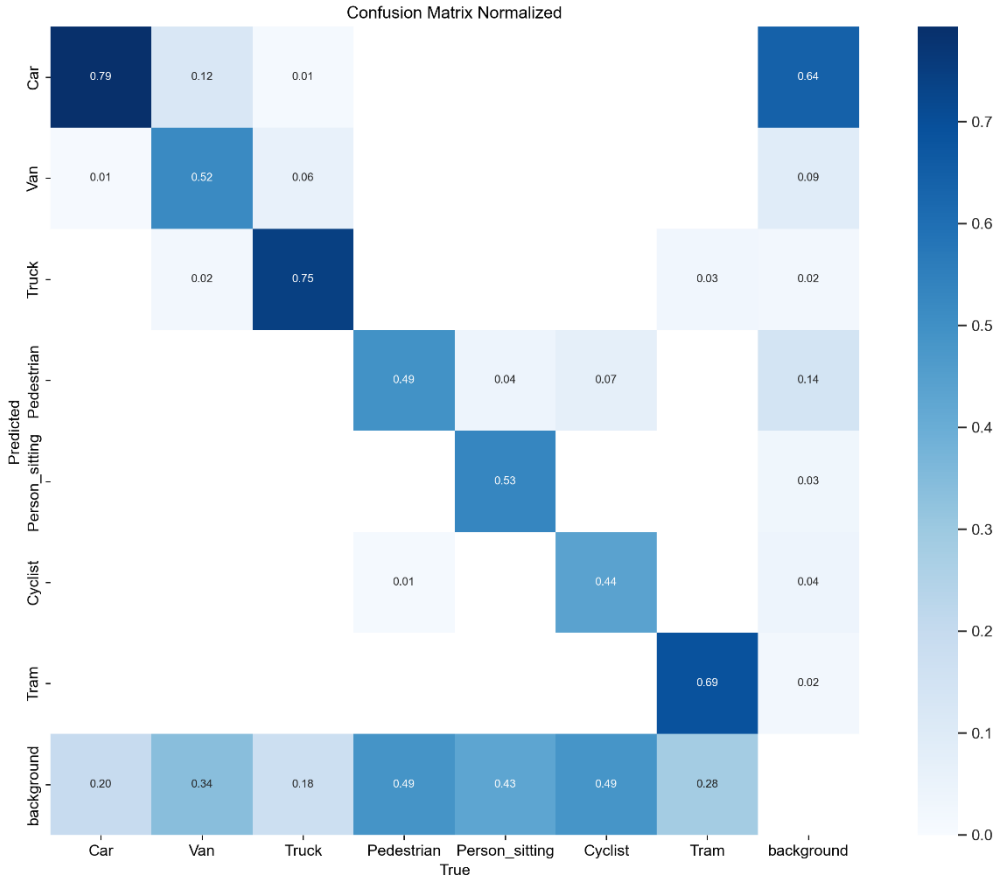
5. Eksperimenti i analiza rezultata

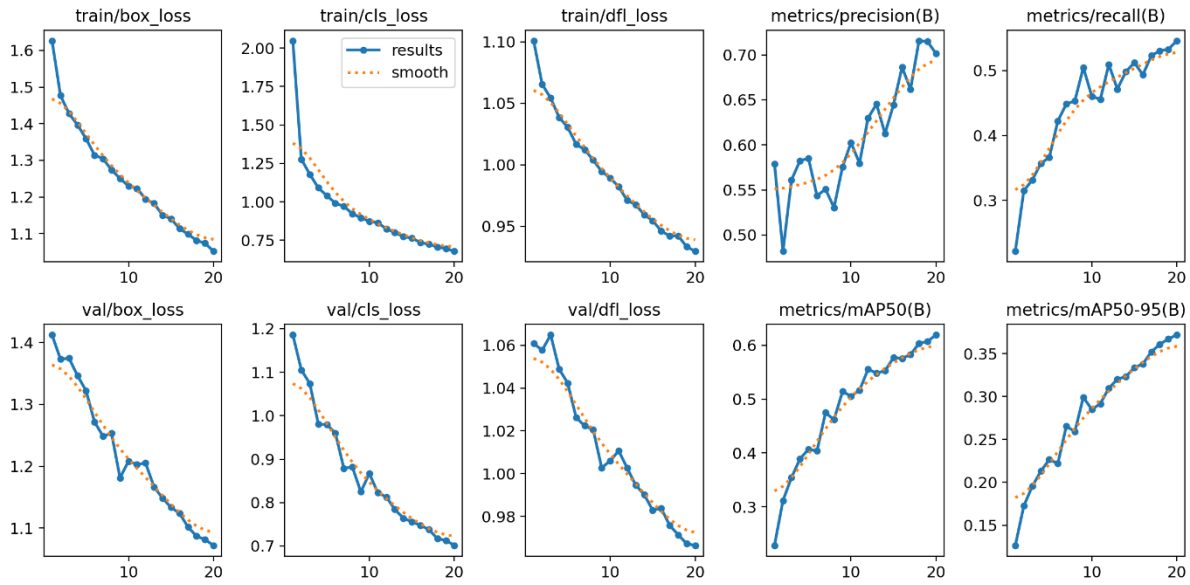
YOLO – početni

Treniranje smo izvršili na 20 epoha s čime smo dobili ove rezultate.

Pri treniranju modela se po matrici konfuzije na slici ispod može vidjeti velika točnost od 80% za detekciju automobila dok i svi ostale detekcije imaju znatnu točnost od ~50% ili više.

Puno klasa van dijagonale matrice nemaju vrijednost što znači da nema puno pogrešnih klasifikacija.





Na slici prije možemo vidjeti metrike treniranja.

1. Analiza Loss-a

- train/box_loss, train/cls_loss, train/dfl_loss, val/box_loss, val/cls_loss, val/dfl_loss
- Sve loss vrijednosti pokazuju opadajući trend, što je pozitivan znak jer znači da model sve bolje uči kako predviđati bounding boxove i klasifikacije.
- Početne vrijednosti su visoke, ali kroz epohe postupno padaju.

Model konvergira ispravno i ne pokazuje znakove prenaučenosti

2. Analiza metričkih performansi

- Preciznost raste, što znači da model postaje bolji u razlikovanju ispravnih od pogrešnih detekcija.
- Recall također raste, što znači da model uspijeva prepoznati više stvarnih objekata u slikama.
- mAP50 raste i približava se 0.8, što znači da model postiže visoku točnost detekcije
- mAP50-95 raste, što pokazuje da model postaje sve precizniji i pri strožim kriterijima

Model poboljšava preciznost detekcije objekata kroz epohe, što je dobar pokazatelj.

Zaključno:

Sve loss vrijednosti opadaju - Model uspješno uči.

Preciznost i recall rastu - Manje pogrešnih detekcija i više pravih pogodaka.

mAP metrike se poboljšavaju - Model je sve precizniji u prepoznavanju objekata.

Na sljedećoj slici se mogu vidjeti primjeri detekcija modela.


```
Ultralytics 8.3.51 🚀 Python-3.9.0 torch-1.12.1+cu113 CUDA:0 (NVIDIA GeForce RTX 2060, 6144MiB)
val: Scanning D:\venv sa python 3.9\rmvid_data\labels\test.cache... 800 images, 0 backgrounds, 0 corrupt: 100%|██████████| 800/800 [00:00<?, ?it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 50/50 [00:04<00:00, 11.98it/s]
                   all        800       4258      0.663      0.565      0.644      0.429
                   Car        728       3124      0.831      0.762      0.845      0.631
                   Van        226        308      0.687      0.584      0.688      0.518
                 Truck        115        120      0.892      0.692      0.823      0.613
            Pedestrian        189        458      0.613      0.402      0.533      0.267
        Person_sitting         11         27      0.341      0.519      0.411      0.221
               Cyclist        119        167      0.563      0.347      0.481      0.278
                  Tram         35         54      0.714      0.648      0.728      0.472
Speed: 0.1ms preprocess, 1.0ms inference, 0.0ms loss, 0.7ms postprocess per image
Results saved to runs\detect\val6
 Evaluacija završena!
 mAP@50: 0.6442
 mAP@50-95: 0.4286
 Precision: 0.6632
 Recall: 0.5649
 F1 Score: 0.6018
```

Prethodna slika prikazuje evaluaciju na testnom skupu slika.

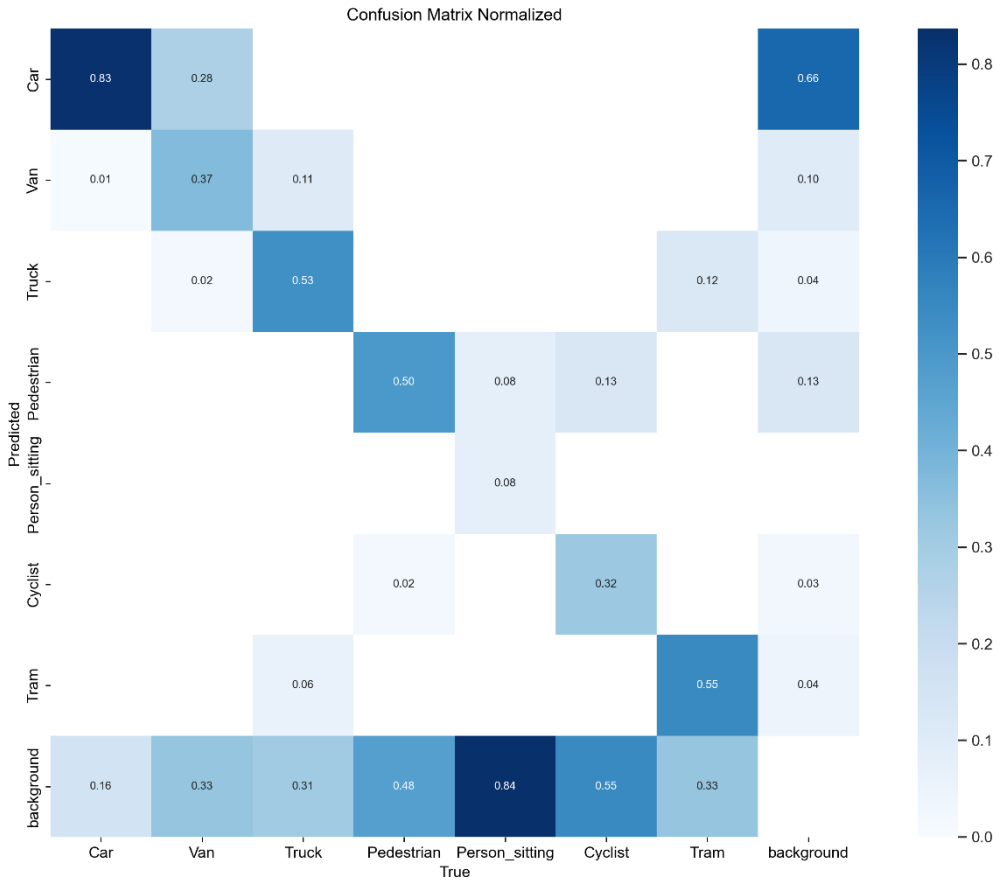
Finalne metrike su :

- mAP@50 0.6422 - model prepoznaje većinu objekata.
- mAP@50-95: 0.4286 - pokazuje da model nije precizan na visokim IoU pragovima
- preciznost 0.6632 - model izbjegava previše lažnih detekcija.
- recall 0.5649 - model propušta dosta pravih objekata.
- F1 score: 0.6018 - Zaključak: Model je umjereno dobar

YOLO - genetski parametri

```
Results saved to D:\yolo_runs\train_kitti44
MLflow: results logged to runs\mlflow
MLflow: disable with 'yolo settings mlflow=False'
 Najbolji model u generaciji 5: ({'batch': 16, 'imgsz': 640, 'learning_rate': 4.581624584296001e-05, 'optimizer': 'Adam'}, 0)

 Genetska optimizacija završena!
 Najbolji pronađeni parametri: {'batch': 16, 'imgsz': 640, 'learning_rate': 4.581624584296001e-05, 'optimizer': 'Adam'}
```

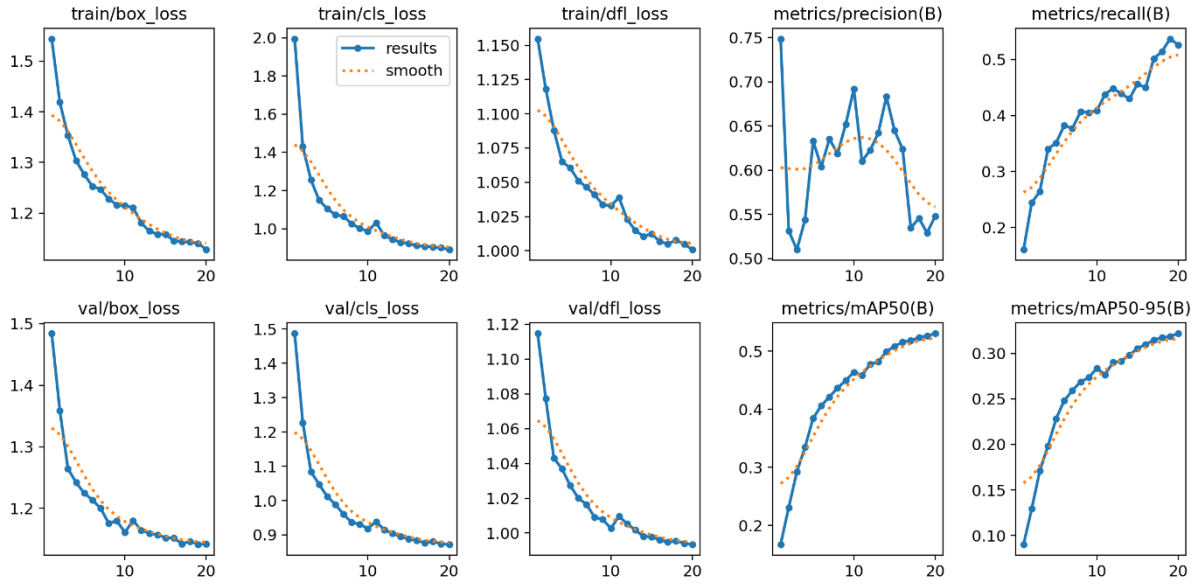
Prethodna slika pokazuje koje parametre je genetski algoritam uzeo za najpovoljniji trening modela. Za razliku od početnog modela velika promjena je povećanje batch size-a i veličine slike.



Treniranje smo izvršili na 20 epoha s čime smo dobili ove rezultate.

Pri treniranju modela se po matrici konfuzije na slici poviše se mogu vidjeti neki lošiji rezultati nego na početnom modelu. Na primjer osoba koja sjedi se jedva prepoznaje dok biciklist i tramvaj se malo bolje prepoznaju.

Usporedno rezultati su lošiji.



Na slici prije možemo vidjeti metrike treniranja.

1. Analiza Loss-a

- train/box_loss, train/cls_loss, train/dfl_loss, val/box_loss, val/cls_loss, val/dfl_loss
- Usporedno s prošlim modelom ove metrike su slične samo što više osciliraju

2. Analiza metričkih performansi

- Preciznost je puno lošija te opada kroz epohe treniranja
- mAP50 i mAP50-95 su slični samo manje osciliraju

Zaključno: genetski model je na kraju lošiji.

Prethodna slika prikazuje rezultate evaluacije modela:

- mAP@50 0.5063 - Početni YOLO model ima znatno viši rezultat, što znači da bolje prepoznaje objekte na nižim IoU pragovima
- mAP@50-95: 0.3120 - Slično kao i za mAP@50,
- preciznost 0.6693 – Genetski model s promijenjenim parametrima ima nešto bolju preciznost, što znači da bolje izbjegava lažne detekcije
- recall 0.3211 - Postoji značajna razlika, genetski optimizirani model pokazuje znatno niži recall
- F1 score: 0.4047 – Genetski model se lošije nosi s pronalaženjem objekata

Fuzzy logika početni i genetski usporedba

```
Klasa: Car
  Visoko povjerenje: 1462 objekata
  Srednje povjerenje: 753 objekata
  Nisko povjerenje: 385 objekata

Klasa: Cyclist
  Visoko povjerenje: 13 objekata
  Srednje povjerenje: 28 objekata
  Nisko povjerenje: 16 objekata

Klasa: Van
  Srednje povjerenje: 75 objekata
  Visoko povjerenje: 87 objekata
  Nisko povjerenje: 36 objekata

Klasa: Pedestrian
  Nisko povjerenje: 78 objekata
  Srednje povjerenje: 161 objekata
  Visoko povjerenje: 11 objekata

Klasa: Truck
  Visoko povjerenje: 55 objekata
  Srednje povjerenje: 23 objekata
  Nisko povjerenje: 5 objekata

Klasa: Tram
  Srednje povjerenje: 18 objekata
  Visoko povjerenje: 17 objekata
  Nisko povjerenje: 6 objekata

Klasa: Person_sitting
  Srednje povjerenje: 12 objekata
  Nisko povjerenje: 7 objekata
```

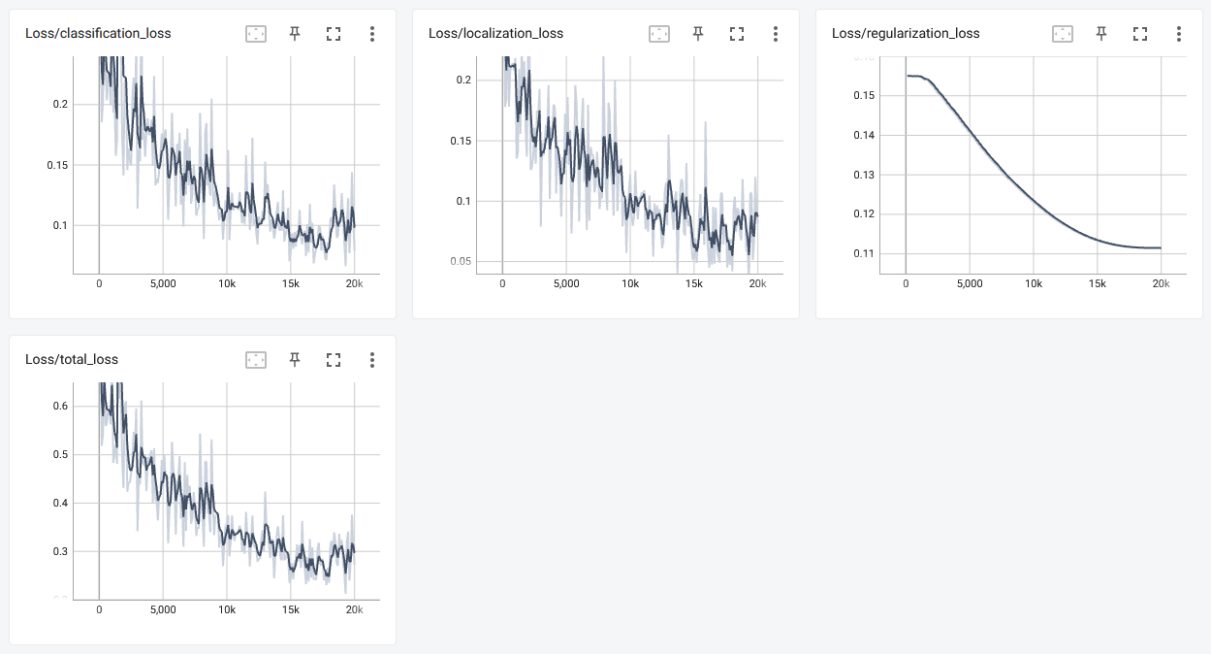
```
Klasa: Car
  Visoko povjerenje: 1729 objekata
  Srednje povjerenje: 915 objekata
  Nisko povjerenje: 500 objekata

Klasa: Cyclist
  Nisko povjerenje: 29 objekata
  Srednje povjerenje: 42 objekata
  Visoko povjerenje: 14 objekata

Klasa: Van
  Visoko povjerenje: 105 objekata
  Nisko povjerenje: 35 objekata
  Srednje povjerenje: 82 objekata

Klasa: Pedestrian
  Srednje povjerenje: 194 objekata
  Nisko povjerenje: 71 objekata
  Visoko povjerenje: 25 objekata

Klasa: Truck
  Visoko povjerenje: 56 objekata
  Srednje povjerenje: 30 objekata
  Nisko povjerenje: 17 objekata

Klasa: Tram
  Nisko povjerenje: 10 objekata
  Srednje povjerenje: 24 objekata
  Visoko povjerenje: 15 objekata

Klasa: Person_sitting
  Srednje povjerenje: 2 objekata
  Nisko povjerenje: 3 objekata
```

Na lijevoj strani prethodne slike se nalaze rezultati fuzzy logike na početnom modelu YOLO, a desno na modelu sa parametrima iz genetskog algoritma

Može se vidjeti da fuzzy logika znatno pomaže genetskom YOLO modelu tako što povećava broj detekcija i viših povjerenja u detekcije

Zaključno fuzzy logika pomaže stabilizirati genetski model te poboljšava rezultate detekcija.

SSD – početni parametri



Na temelju prikazanih grafova može se analizirati proces učenja modela kroz različite komponente loss-a:

1. Classification Loss

    Gornji lijevi graf prikazuje gubitak klasifikacije

    Model postupno uči razlikovati klase objekata

2. Localization Loss

    Gornji srednji graf prikazuje lokalizacijski gubitak, tj. sposobnost modela da točno postavi bounding boxove oko objekata.

    Model postupno poboljšava pozicioniranje objekata

3. Regularization Loss

Regularizacija učinkovito djeluje, što znači da model postaje stabilniji tijekom učenja.

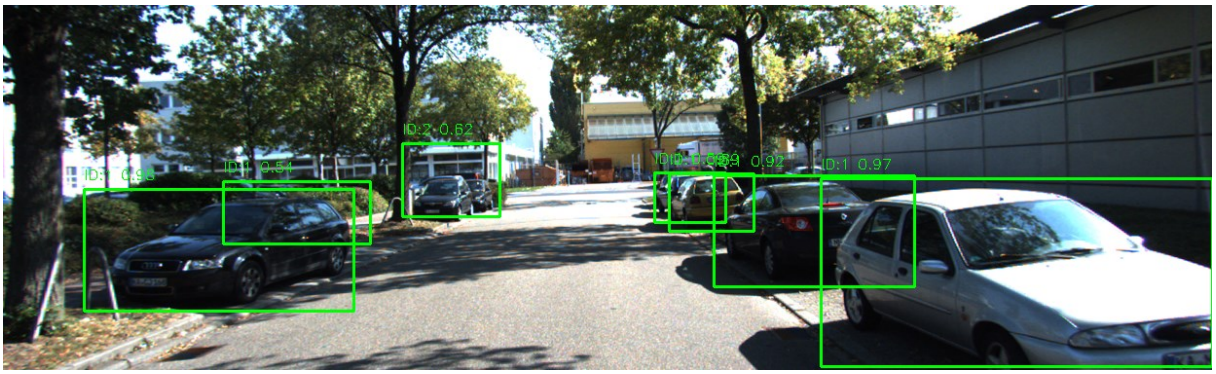
4. Total Loss

Unatoč oscilacijama u lokalizacijskom i klasifikacijskom loss-u, ukupni gubitak se smanjuje, što znači da model uči na zadovoljavajući način.

Zaključak:

Model uspješno uči, s jasnim trendom smanjenja svih komponenti gubitka.

Sljedeća slika je prikaz detekcije objekata u modelu




```
creating index...
index created!
creating index...
index created!
Running per image evaluation...
Evaluate annotation type *bbox*
DONE (t=7.69s).
Accumulating evaluation results...
DONE (t=1.41s).
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.379
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.662
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.384
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.142
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.358
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.532
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.299
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.497
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.529
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.356
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.498
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.675
[INFO] Evaluacija gotova.
Evaluacija završena!
```

Prethodna slika prikazuje rezultate evaluacije modela na testnom skupu.

1. Average Precision

(IoU=0.50:0.95) = 0.379 - model postiže solidnu točnost

(IoU=0.50) = 0.662 - Visoka preciznost na nižem pragu preklapanja – model dobro prepoznaje objekte, ali ponekad griješi u preciznom pozicioniranju bounding boxova.

(IoU=0.75) = 0.384 - Model se teže prilagođava strožim kriterijima preklapanja

Po veličini objekata:

Mali objekti (small) = 0.142 - izazovno za model zbog manje vidljivosti.

Srednji objekti (medium) = 0.358 - solidni rezultati.

Veliki objekti (large) = 0.532 - najbolji rezultati, što je očekivano jer su veći objekti vizualno istaknutiji.

2. Average Recall

mjeri koliko je model uspješan u pronalasku svih relevantnih objekata.

AR (IoU=0.50:0.95, maxDets=1) = 0.299

AR (maxDets=10) = 0.497 Značajno poboljšanje s više detekcija.

AR (maxDets=100) = 0.529 Sveukupno solidan rezultat.

AR po veličini objekata:

Mali objekti (small) = 0.356 - relativno nizak zbog složenosti detekcije manjih objekata.

Srednji objekti (medium) = 0.498 - dobra učinkovitost.

Veliki objekti (large) = 0.675 – dobar rezultat

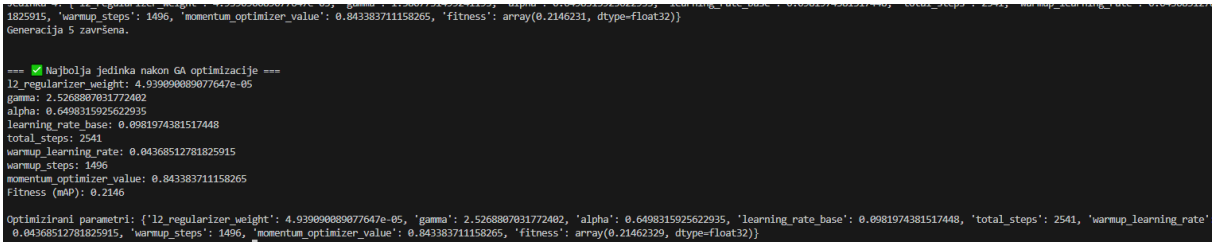
Zaključak:

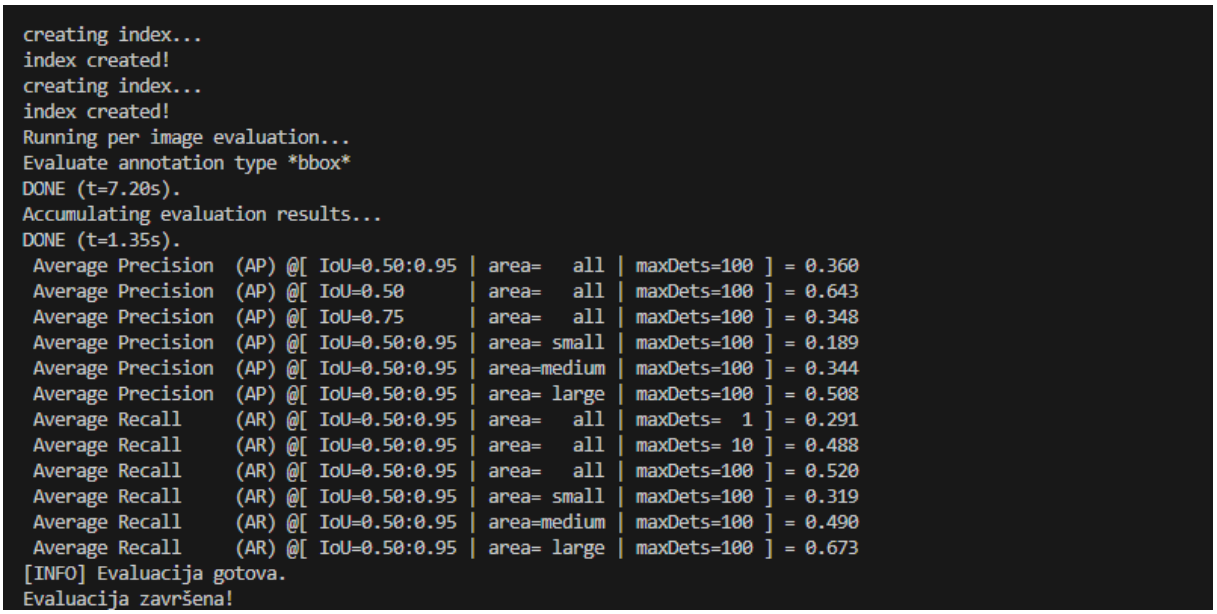
Učinkovitije detektira velike objekte u odnosu na male objekte, što sugerira potrebu za dodatnom augmentacijom podataka i finim podešavanjem za manje objekte.

Precision na 0.50 IoU (0.662) ukazuje na to da model dobro prepoznaje objekte, ali treba poboljšanja u preciznijem pozicioniranju.

SSD – genetski parametri

Sljedeća slika prikazuje output odabiranja najbolje jedinke genetskog algoritma.





Prethodna slika prikazuje rezultate evaluacije genetskog modela na testnom skupu

Usporedno s prethodnim rezultatima početnog modela:

Average Precision

AP (IoU=0.50:0.95) = 0.360 - Blagi pad točnosti

AP (IoU=0.50) = 0.643 - Lagani pad preciznosti na nižem pragu.

AP (IoU=0.75) = 0.348 - Smanjena preciznost kod strožih kriterija preklapanja.

Precision po veličini objekata:

Mali objekti = 0.189 – Poboljšanje – model sada bolje detektira male objekte.

Srednji objekti = 0.344 - Blagi pad detekcije srednjih objekata.

Veliki objekti = 0.508 - Smanjena točnost za velike objekte.

Average Recall

AR (IoU=0.50:0.95, maxDets=1) = 0.291 - niže

AR (maxDets=10) = 0.488 - niže

AR (maxDets=100) = 0.520 - niže

AR po veličini objekata:

Mali objekti (small) = 0.319 - niže

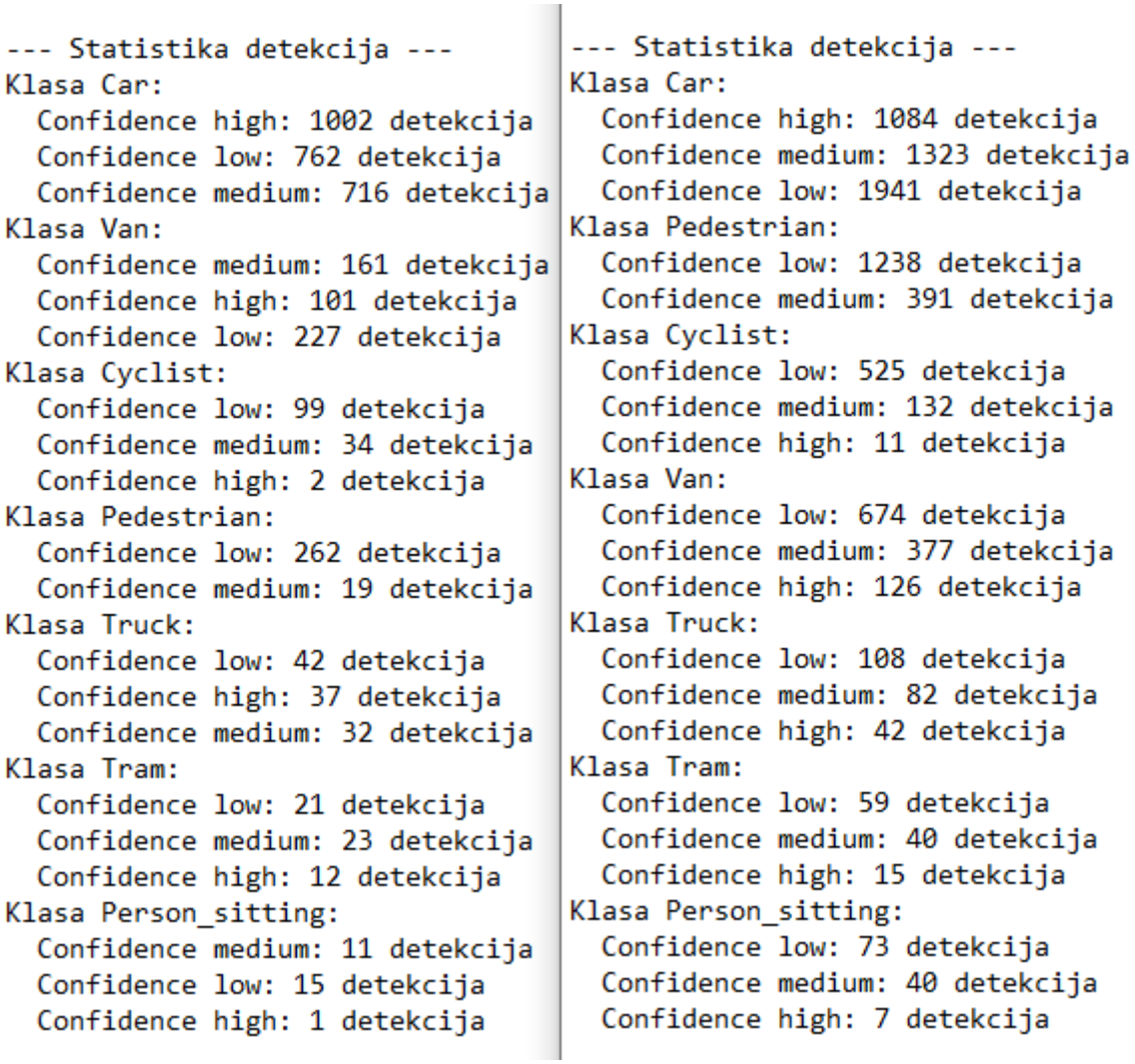
Srednji objekti (medium) = 0.490 - niže

Veliki objekti (large) = 0.673 – slično

Zaključak:

- Pad ukupne preciznosti
- Vjerojatno uzrokovan smanjenom učinkovitošću na srednjim i velikim objektima.
- Napredak u detekciji malih objekata
- Pozitivno poboljšanje, što može ukazivati na bolje prilagođavanje modela za sitne objekte.

Fuzzy usporedba SSD početni i genetski parametri



```
--- Statistika detekcija ---
Klasa Car:
  Confidence high: 1002 detekcija
  Confidence low: 762 detekcija
  Confidence medium: 716 detekcija
Klasa Van:
  Confidence medium: 161 detekcija
  Confidence high: 101 detekcija
  Confidence low: 227 detekcija
Klasa Cyclist:
  Confidence low: 99 detekcija
  Confidence medium: 34 detekcija
  Confidence high: 2 detekcija
Klasa Pedestrian:
  Confidence low: 262 detekcija
  Confidence medium: 19 detekcija
Klasa Truck:
  Confidence low: 42 detekcija
  Confidence high: 37 detekcija
  Confidence medium: 32 detekcija
Klasa Tram:
  Confidence low: 21 detekcija
  Confidence medium: 23 detekcija
  Confidence high: 12 detekcija
Klasa Person_sitting:
  Confidence medium: 11 detekcija
  Confidence low: 15 detekcija
  Confidence high: 1 detekcija
```

```
--- Statistika detekcija ---
Klasa Car:
  Confidence high: 1084 detekcija
  Confidence medium: 1323 detekcija
  Confidence low: 1941 detekcija
Klasa Pedestrian:
  Confidence low: 1238 detekcija
  Confidence medium: 391 detekcija
Klasa Cyclist:
  Confidence low: 525 detekcija
  Confidence medium: 132 detekcija
  Confidence high: 11 detekcija
Klasa Van:
  Confidence low: 674 detekcija
  Confidence medium: 377 detekcija
  Confidence high: 126 detekcija
Klasa Truck:
  Confidence low: 108 detekcija
  Confidence medium: 82 detekcija
  Confidence high: 42 detekcija
Klasa Tram:
  Confidence low: 59 detekcija
  Confidence medium: 40 detekcija
  Confidence high: 15 detekcija
Klasa Person_sitting:
  Confidence low: 73 detekcija
  Confidence medium: 40 detekcija
  Confidence high: 7 detekcija
```

Na lijevoj strani prethodne slike se nalaze rezultati fuzzy logike na početnom modelu SSD, a desno na modelu sa parametrima iz genetskog algoritma

Može se vidjeti da fuzzy logika znatno pomaže genetskom SSD modelu tako što povećava broj low detekcija i ponekad high povjerenja u detekcije

Zaključno fuzzy logika pomaže stabilizirati genetski model te poboljšava rezultate detekcija

6. Zaključak

U ovom radu istražili smo različite metode detekcije objekata na cesti za autonomna vozila, koristeći modele YOLOv8 i SSD. KITTI dataset korišten je kao osnova za treniranje i evaluaciju modela, skup podataka koji simulira stvarne uvjete cestovnog prometa. Glavni cilj rada bio je analizirati performanse ovih modela te istražiti utjecaj genetskog algoritma i fuzzy logike na njihovu učinkovitost.

Rezultati eksperimenata pokazali su da YOLOv8 model postiže bolje rezultate u usporedbi sa SSD modelom, posebno u pogledu preciznosti i brzine detekcije.

Međutim, primjena genetskog algoritma za optimizaciju hiperparametara nije uvijek dovodila do poboljšanja performansi. Kod YOLOv8 modela, genetski optimizirani parametri rezultirali su lošijim rezultatima u odnosu na početni model, dok je kod SSD modela došlo do blagog poboljšanja u detekciji malih objekata, ali uz pad ukupne preciznosti.

Fuzzy logika pokazala se korisnom u poboljšanju pouzdanosti detekcija, posebno kod genetski optimiziranih modela, gdje je pomogla u stabilizaciji rezultata i povećanju broja detekcija s visokim povjerenjem.