FAKULTET ELEKTROTEHNIKE, STROJARSTVA I BRODOGRADNJE SPLIT

Upravljanje lukom

Računska inteligencija - seminarski rad

Andrej Vidović
Toni Jakovčević
Duje Srhoj
Ivan Dagelić
Bruno Grbavac

Diplomski studij računarstva (250)
Akademska godina 2021./22.

Sadržaj

1 Uvod

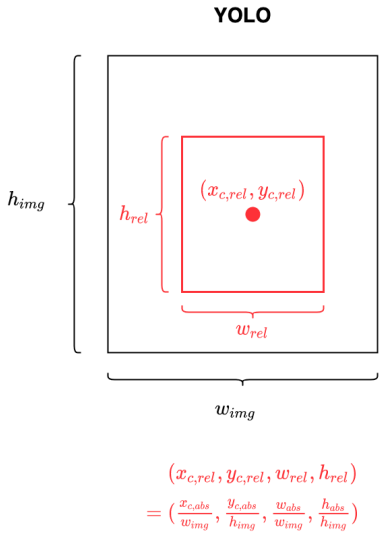
Sustav za nadzor luke kao zadaću ima prepoznavanje i lociranje plovila unutar prostora luke. Koristeći ustaljenu terminologiju, sam model vrši zadaće **lokalizacije** (određivanja položaja) i **višeklasne klasifikacije** plovila.

Plovila se klasificiraju u sljedeće klase: **kruzer, trajekt, katamaran, tegljač/koća, jahta, jedrilica, gliser, jetski, glider** i **ostalo**.

Izvorni skup podataka (eng. *dataset*) sadrži **214 slika** Trajektne luke Split u formatu **1920x1080** dubine **24 bita**.

2 Labeliranje

Fotografije su labelirane koristeći alat **labelImg**. Same oznake pohranjene su u **YOLO formatu** koji sadrži cjelobrojnu oznaku klase, vrijednost koordinate centralne točke bounding box-a na koordinatnoj osi apcisa, vrijednost koordinate centralne točke bounding box-a na koordinatnoj osi ordinata, širinu bounding box-a relativno ukupnoj širini slike i visinu bounding box-a relativno ukupnoj visini slike.



Slika 1: YOLO format labeliranja.

```
1 kruzer
2 trajekt
3 katamaran
4 tegljac_koca
5 jahta
6 jedrilica
7 gliser
8 jetski
9 glider
10 ostalo
```

Slika 2: Cjelobrojne oznake klasa i pripadne klase.

```
1 0.056510 0.651389 0.111979 0.069444
3 0.136198 0.637963 0.027604 0.033333
1 0.173437 0.616667 0.088542 0.059259
1 0.107552 0.749074 0.180729 0.142593
1 0.182292 0.712037 0.129167 0.079630
9 0.902865 0.568056 0.005729 0.006481
2 0.547135 0.711574 0.029687 0.056481
2 0.554427 0.687963 0.034896 0.044444
9 0.187240 0.795370 0.008854 0.012963
```

Slika 3: YOLO format labeliranja.

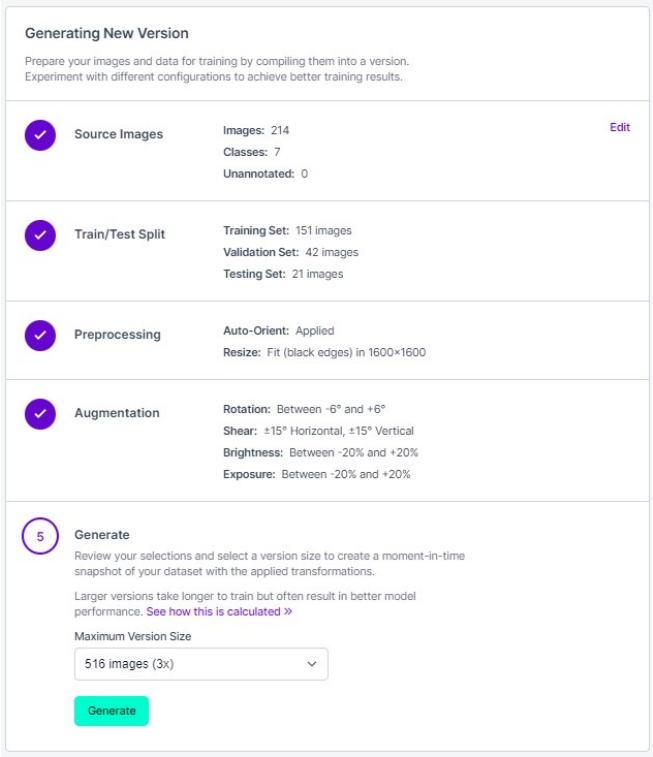
3 Predobrada podataka

Dataset se prije korištenja za treniranje modela podliježe raznim procesima obrade podataka, to jest nad podacima se vrši eng. *preprocessing*. Za samu obradu i upravljanje podacima korištena je platforma **Roboflow**.
Prethodno labelirani podaci podignuti su na Roboflow gdje je zatim izvršena njihova podjela u podatke za treniranje, validaciju i testiranje. Sama podjela vrši se nasumično pri čemu se **70%** slika koristi za treniranje, **20%** za validaciju dok se preostalih **10%** koristi za testiranje dobivenoga modela.

Prva promjena je preoblikovanje fotografija na veličinu **1600x1600** piksela, uz dodavanje crnog prostora.
Nadalje, besplatni plan Roboflow platforme omogućava **utrostručenje** broja slika za treniranje modela koristeći razne tehnike proširenja (eng. *augmentation*). Proširenje se radi kako bi se povećala moć generalizacije modela (izbjeći pretreniranost), tako da generiranim podacima (slikama) predvidimo moguća stanja okoliša u kojima se prikupljaju podaci, npr. različiti kutovi, smanjena svjetlost itd., koja nisu zastupljena u izvornom skupu podataka.

- **Rotacija** (eng. *rotation*) - slike su rotirane u rasponu od -6° do +6°
- **Svjetlina** (eng. *brightness*) - slikama se mijenja svjetlina u rasponu od -20% do +20%
- **Ekspozicija** (eng. *exposure*) - slikama se mijenja ekspozicija u rasponu od -20% do +20%
- **Iskrivljenje** (eng. *shear*) - slike se iskrivljuju pri kutu u rasponu od -15° do +15° horizontalno i vertikalno

Ovim procesom veličina skupa podataka povećana je sa **214** na **516** slika trajektne luke.



Slika 4: Roboflow sučelje za predobradu.

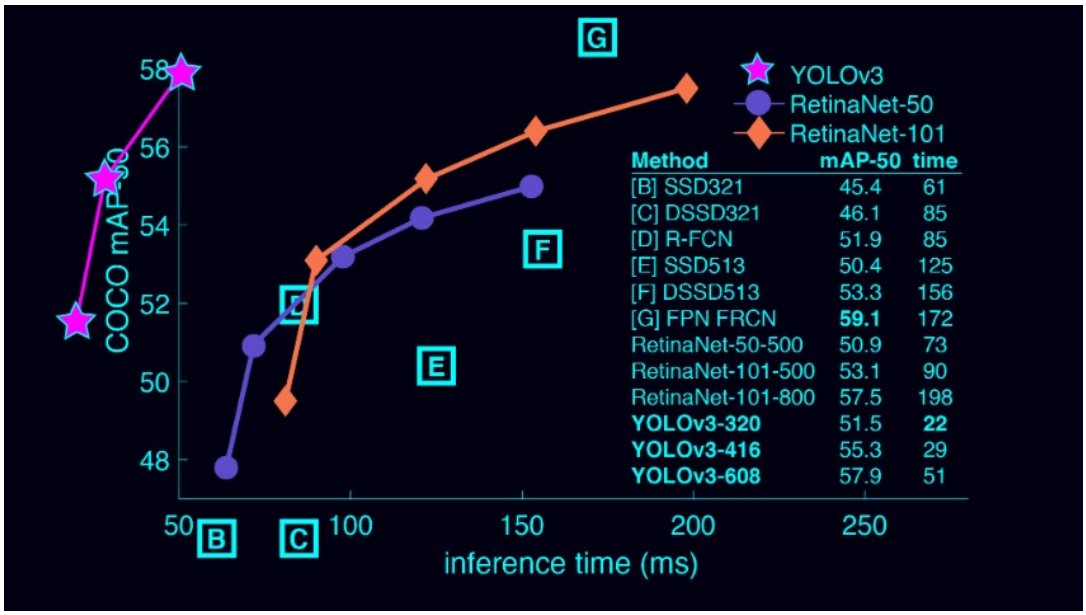
Slika 5: Primjer slike dobivene augmentacijom dataseta.

Slika 6: Mozaik slika za treniranje modela.

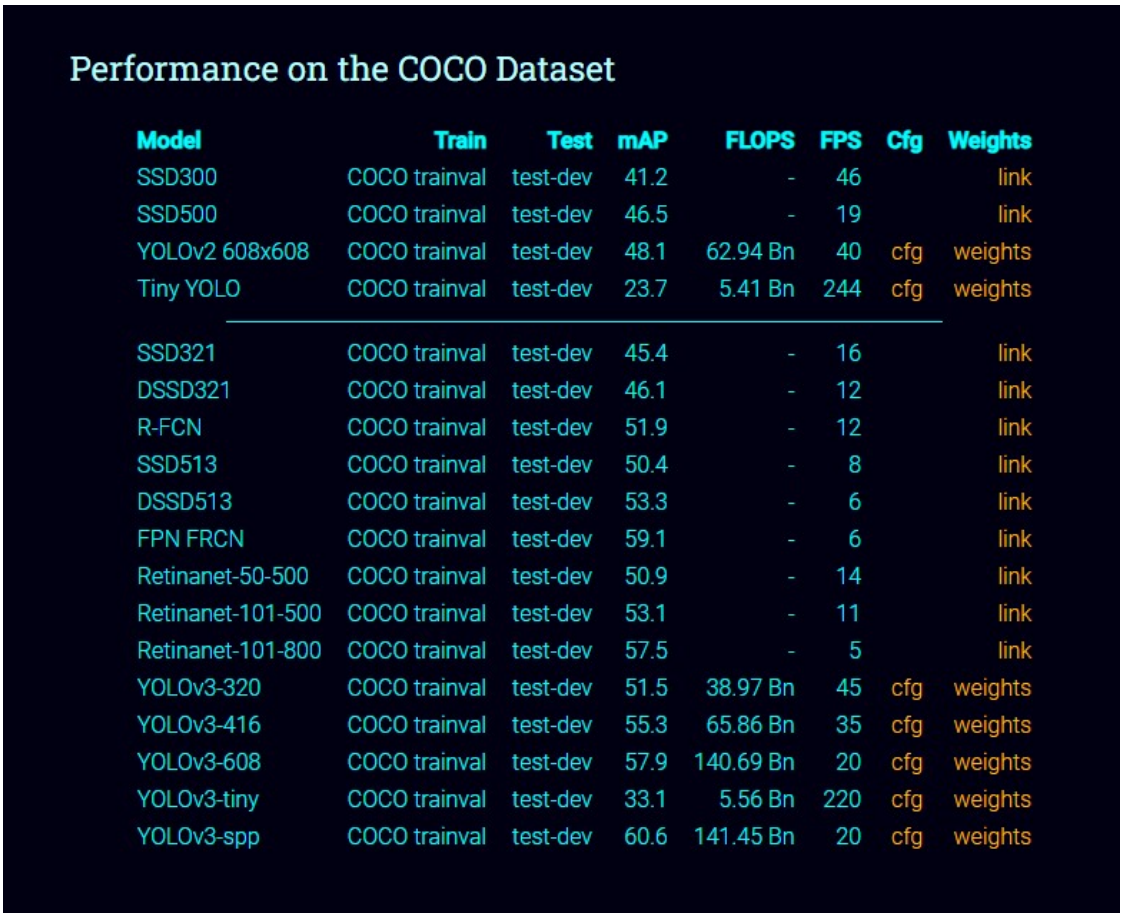
4 YOLO

YOLO je konvolucijska neuralna mreža koju je 2015. godine predstavio **Joseph Redmon** u svom radu *You Only Look Once: Unified, Real-Time Object Detection*. Prethodni modeli, korišteni za detekciju objekata na slikama, preinaka su običnih klasifikatorskih modela koji su uzastopnim primjenjivanjem na ulaznu sliku (sliding window shema) koristili za detekciju. Primjena na različitim dijelovima slike dala bi stoga područja visoke i niske vjerojatnosti postojanja objekta, koja bi zatim ovisno o kriterijima bila proglašena detekcijama ili ne. Medium YOLO koristi drukčiji pristup. Jedna neuralna mreža primjenjuje se na čitavu sliku (ne dijelove), mreža sliku dijeli u regije i predviđa obuhvatne okvire objekata (*eng.* bounding box) i vjerojatnosti detekcije za svaku regiju.[2]

Ovakav pristup ima više prednosti u odnosu na tradicionalan pristup sa preinačenim klasifikatorima. Činjenicom što se primjenjuje na cijeloj slici, predviđanja u YOLO mreži se donose na osnovu globalnog konteksta - čitave slike. Također, činjenicom što nije potrebno algoritam pokretati tisuće puta (broj ovisi o slici i očekivanim rezultatima), YOLO je i puno brža mreža od prethodno ustaljenih **R-CNN (1000x brža)** i **Fast R-CNN (100x brža)** mreža.[2]



Slika 7: Usporedba ovisnosti mAP metrike i vremena zaključivanja. [2]



Performance on the COCO Dataset

Model	Train	Test	mAP	FLOPS	FPS	Cfg	Weights
SSD300	COCO trainval	test-dev	41.2	-	46		link
SSD500	COCO trainval	test-dev	46.5	-	19		link
YOLOv2 608x608	COCO trainval	test-dev	48.1	62.94 Bn	40	cfg	weights
Tiny YOLO	COCO trainval	test-dev	23.7	5.41 Bn	244	cfg	weights
SSD321	COCO trainval	test-dev	45.4	-	16		link
DSSD321	COCO trainval	test-dev	46.1	-	12		link
R-FCN	COCO trainval	test-dev	51.9	-	12		link
SSD513	COCO trainval	test-dev	50.4	-	8		link
DSSD513	COCO trainval	test-dev	53.3	-	6		link
FPN FRCN	COCO trainval	test-dev	59.1	-	6		link
Retinanet-50-500	COCO trainval	test-dev	50.9	-	14		link
Retinanet-101-500	COCO trainval	test-dev	53.1	-	11		link
Retinanet-101-800	COCO trainval	test-dev	57.5	-	5		link
YOLOv3-320	COCO trainval	test-dev	51.5	38.97 Bn	45	cfg	weights
YOLOv3-416	COCO trainval	test-dev	55.3	65.86 Bn	35	cfg	weights
YOLOv3-608	COCO trainval	test-dev	57.9	140.69 Bn	20	cfg	weights
YOLOv3-tiny	COCO trainval	test-dev	33.1	5.56 Bn	220	cfg	weights
YOLOv3-spp	COCO trainval	test-dev	60.6	141.45 Bn	20	cfg	weights

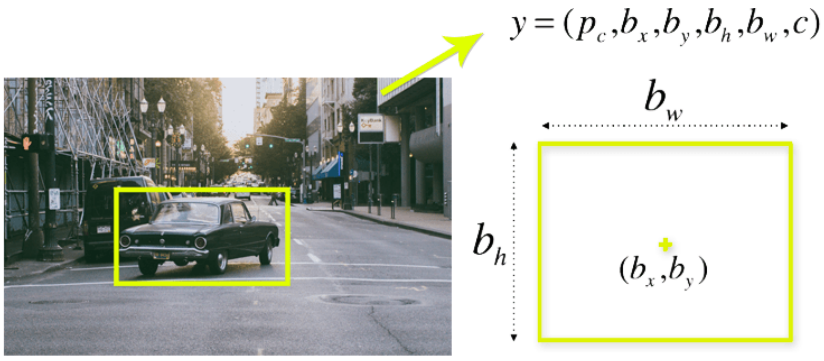
Slika 8: Usporedba performansi mreža na COCO skupu podataka.[2]

4.1 Temelji rada YOLO mreže

- **Residual blocks** - slika se dijeli na polja dimenzija SxS. Svako polje detektira objekte koji se nalaze unutar koristeći pritom i informacije iz globalnog konteksta. [1]

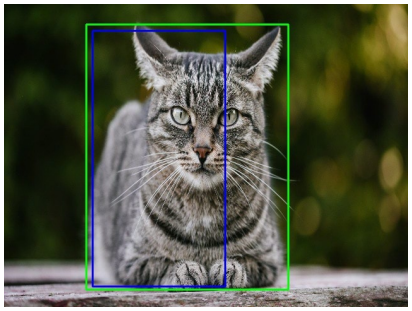
Slika 9: Podjela slike na ćelije. [1]

- **Regresija obuhvatnog okvira** (*eng.* bounding box regression) - okvir je obris koji obuhvaća detektirani objekt. Za obris je stoga potrebno poznavati **širinu (bw)**, **visinu (bh)**, **klasu detektiranog objekta(c)** i **središte okvira (bx, by)**. *Single bounding box regression* je postupak koji YOLO koristi za određivanje vjerojatnosti detektiranog objekta i njegovog okvira (uključujući sve gore navedene parametre). [1]



Slika 10: Podjela slike na ćelije. [1]

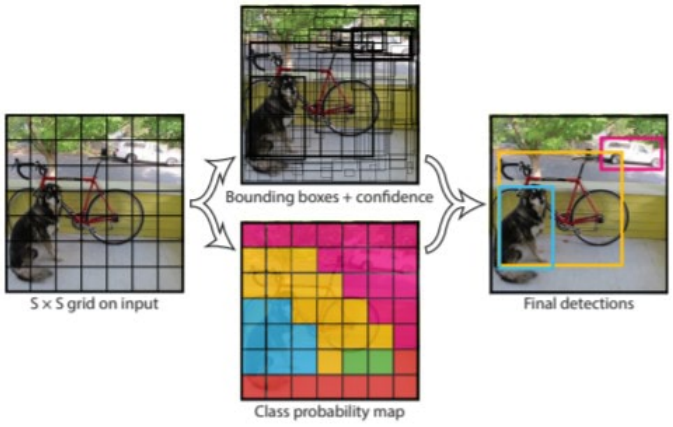
- **IOU - presjek po uniji** (*eng. intersection over union* je metrika koja opisuje točnost preklapanja stvarnog (*eng.* ground truth) i izračunatog okvira. Korištenje IOU osigurava što savršenije obuhvaćanje objekta okvirom. [1]



Slika 11: Plavo - predviđeni, zeleno - stvarni okvir. [1]

U gornjem primjeru vidimo predviđeni okvir koji pokriva cca. 70% stvarnog okvira.

U suštini, slika se dijeli na ćelije grida. Svaka ćelija potom predviđa obuhvatne okvire objekata s pripadajućom ocjenom pouzdanosti (*eng.* confidence score). Uz okvire, ćelije predviđaju i klase objekata. Predviđanja svih objekata istovremeno (u jednoj rundi) 1 konvolucijska mreža. Korištenjem IOU zatim se eliminiraju okviri koji ne odgovaraju karakteristikama traženih objekata te bi stoga rezultantni okviri trebali odlično pristajati. [1]



Slika 12: Proces rada YOLO mreže. [1]

5 YOLO V5

Pet godina nakon originalne mreže, 18. svibnja 2020. **Glen Jocher** objavio je YOLOv5 mrežu temeljenu na **PyTorch frameworku**. Kao i njegovi prethodnici YOLOv5 je *"single stage"* detektor objekata.Mreža se sastoji od tri glavna djela: **okosnice**, **vrata** i **glave** modela. [4]

Okosnica (*eng.* backbone) ima kao glavnu zadaću iz ulazne slike ekstrahirati bitne značajke (*eng.* features). Za ovu svrhu YOLOv5 koristi *"Cross Stage Partial"* mreže kako bi se izvukle što informativnije značajke, osim informativnosti značajki CSPNet donosi i značajna poboljšanja što se tiče vremena potrebnog za procesiranje.. [4]

Vrat modela u YOLOv5 generira **piramide značajki** (*eng.* feature pyramids). Glavna je zadaća navedenih piramida poboljšati generalizaciju modela pri skaliranju ciljanih objekata. Piramide značajki jako su korisne jer omogućuju da model sačuva performanse i na objektima koji mu nisu predočeni, to jest da funkcionira i na objektima u različitim veličinama od onih iz podataka za treniranje. Postoje različiti modeli koji implementiraju piramide značajki poput FPN, BiFPN mreža, a sam YOLOv5 za svoj vrat koristi **PANet**.. [4]

Glava modela u YOLOv5 ne razlikuje se od onih u trećoj i četvrtoj inačici YOLO-a. Glava kao glavnu zadaću (:D) ima primjenu **anchor boxova** nad ekstrahiranim značajkama i generiranje izlaza - vektora vjerojatnosti pripadnosti klasama, okvirima i *"objectness"* ocjenom. Izbor aktivacijske funkcijske funkcije jedna je od glavnih značajki neuralnih mreža, YOLOv5 koristi **Leaky ReLU** u skrivenim slojevima mreže dok posljednji, detekcijski sloj koristi **sigmoid** kao aktivacijsku funkciju.. [4]

5.1 Verzije YOLOv5

Paket YOLOv5 donosi pet različitih modela različite veličine [3]:

- **YOLOv5n - nano** model je najnovija prinova v5 modela. Ciljana uporaba ove inačice je primjena u IoT uređajima. Zauzima manje od 2.5 MB u INT8 ili oko 4 MB u FP32 formatu, što ga čini jako prikladnim za navedene mobilne primjene.
- **YOLOv5s - small** model sa 7.2 milijuna podesivih parametara za ciljanu primjenu ima sustave sa potrebom za brzim zaključivanjem ili resursima ograničene sustave (npr. izvršavanje na CPU)
- **YOLOv5m - medium** sadrži 21.2 milijuna parametara te se nudi kao idealan kompromis tj. rješenje u većini primjena zbog ravnoteže između brzine zaključivanja i rezultantne preciznosti.

- **YOLOv5l - large** sa 46.5 milijuna parametara za treniranje idealna je opcija za skupove podataka sa sitnim objektama čiji pripadajući sustavi ne ovise nužno o brzini zaključivanja.
- **YOLOv5x - extra large** model je najveći u obitelji modela i stoga nosi najveću preciznost. Sadrži 86.7 milijuna parametara za treniranje.

Model Name	Params (Million)	Accuracy (mAP 0.5)	CPU Time (ms)	GPU Time (ms)
YOLOv5n	1.9	45.7	45	6.3
YOLOv5s	7.2	56.8	98	6.4
YOLOv5m	21.2	64.1	224	8.2
YOLOv5l	46.5	67.3	430	10.1
YOLOv5x	86.7	68.9	766	12.1

Slika 13: Usporedba različitih YOLOv5 modela. [3]

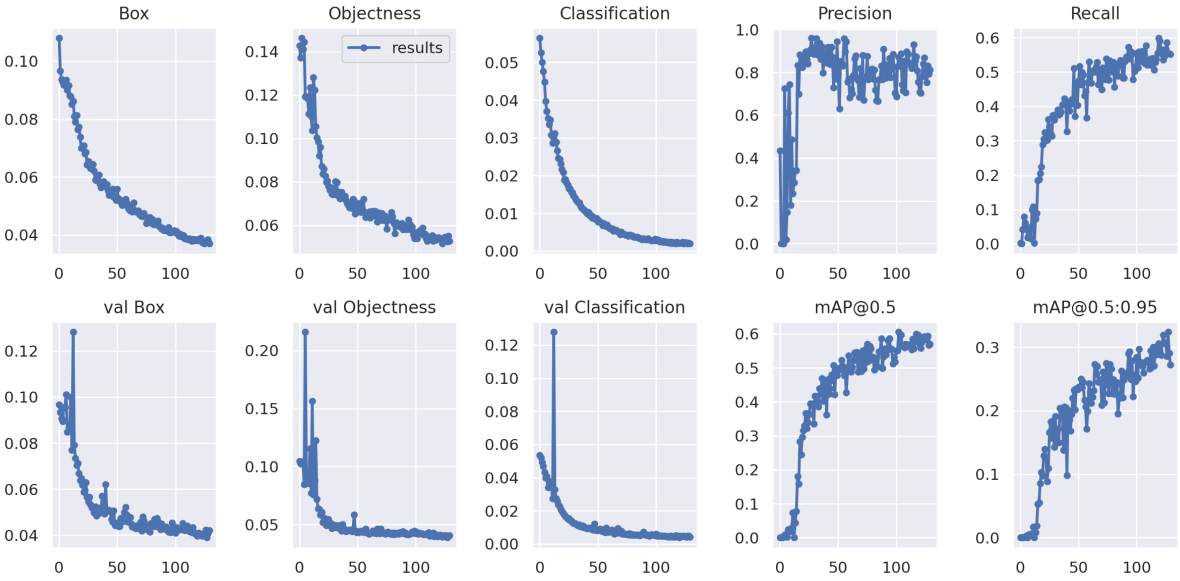
6 Model za detekciju plovila u T.L. Split temeljen na YOLOv5

U sklopu ovog seminara na već navedenom skupu podataka trenirana su, na istim grafičkim karticama - NVIDIA Tesla T4, za usporedbu, 2 modela iz obitelji YOLOv5. Trenirani su 2. najmanja i 2. najveća verzija mreže, to jest YOLOv5 **small** i **large**.

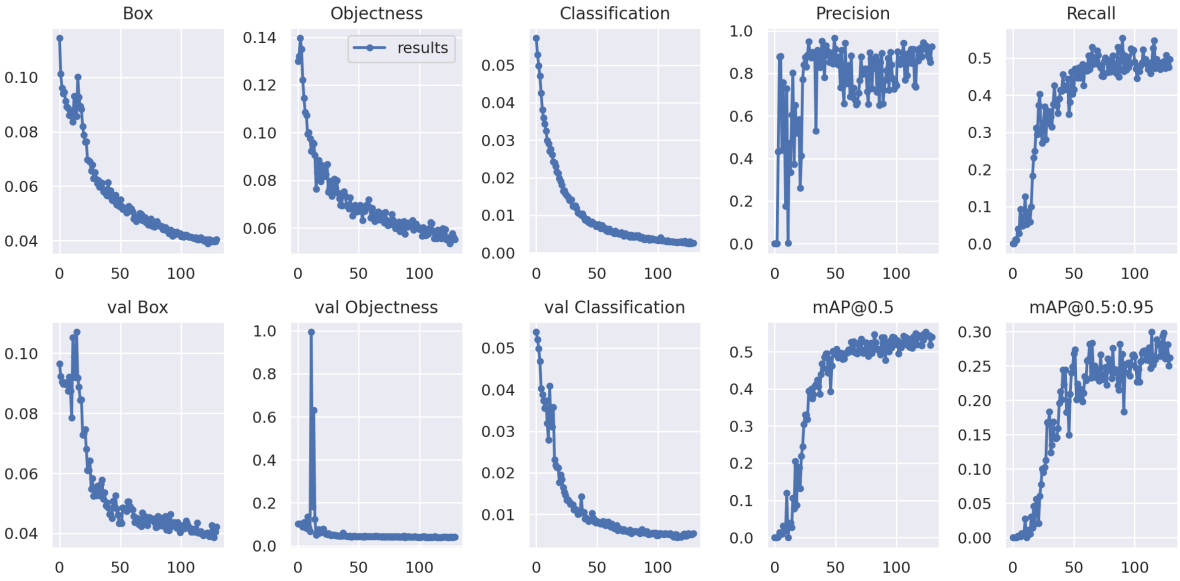
Metrike treniranja koje YOLOv5 samostalno prikuplja prikazane su pomoću TensorBoard biblioteke. Kroz par pokušaja treniranja YOLOv5l na **TensorBoard** grafovima je bilo vidljivo da će poboljšanje mreže stagnirati na oko 130+ epoha, stoga bi nam tu već prijetila pretreniranost te bespotrebno traćenje vremena i resursa. Stoga je, iako je zbog manje parametara očekivano na manjoj mreži postići nekakav oblik stagnacije poboljšanja i pretreniranosti, isti broj epoha korišten i na manjoj mreži.

Slika 14: Prikaz TensorBoard sučelja u Colabu.

Težine i *"anchor boxovi"* samog modela nisu mijenjati, iako je prilagođenost *anchora* podacima jako bitna zbog detekcije objekata ciljanih veličina, YOLOv5 sam izvršava ***AutoAnchor*** koji provjerava prilagođenost podacima i u slučaju neprilagođenosti sam računa nove *"anchor boxove"*. [5]



Slika 15: Metrike Large modela.



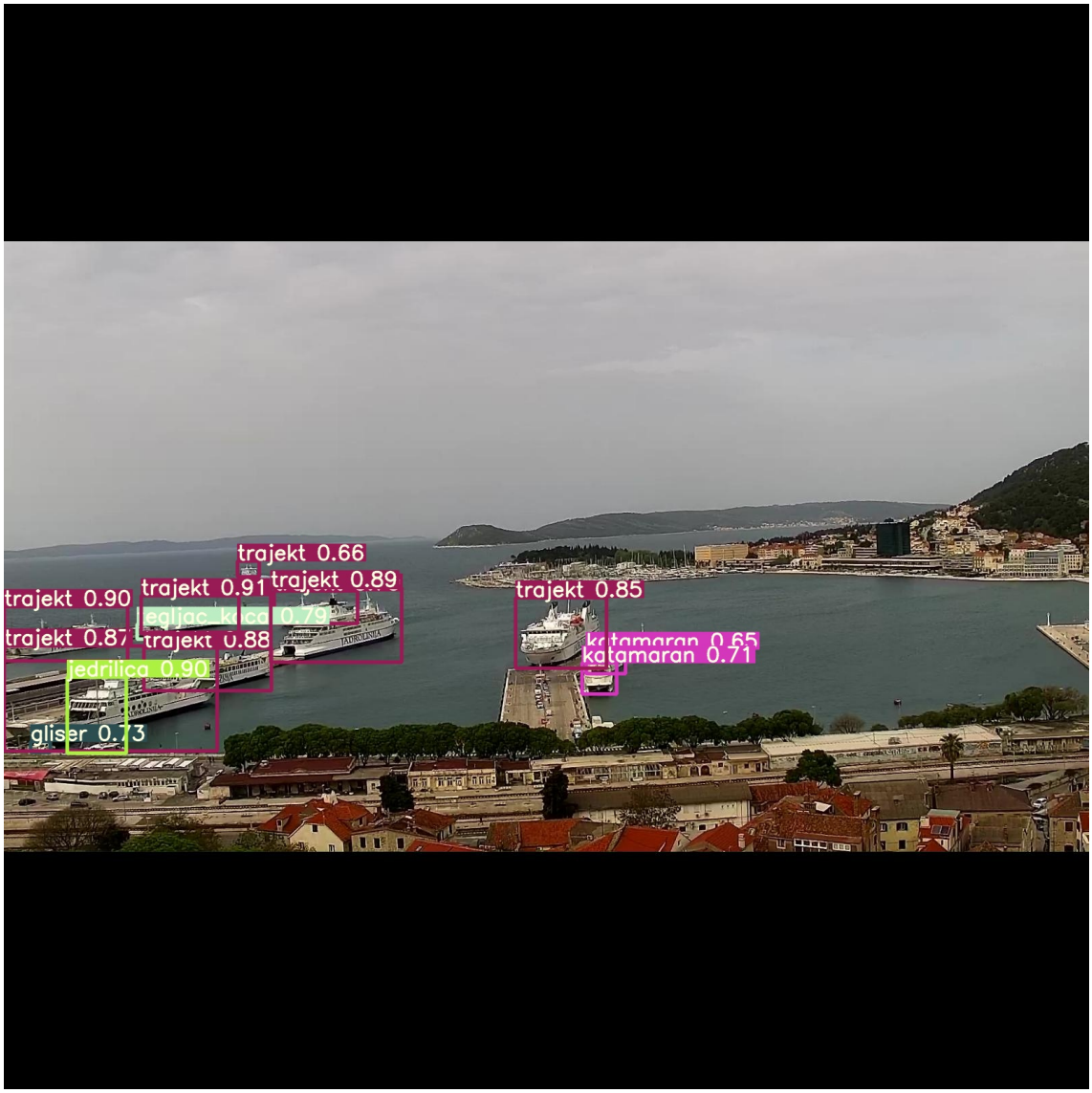
Slika 16: Metrike Small modela.

Sama usporedba rezultata dobivenih ovim modelima može se svesti na sljedeću tablicu:

	YOLOv5 Small	**YOLOv5 Large**
Vrijeme po epohi treniranja	32 s	1 min 43 s
Ukupno vrijeme treniranja	1h 17 min 47 s	3h 59 min 8s
mAP@0.5	0.553	0.592
Preciznost	0.931	0.869
Odziv	0.491	0.547
Objectness	0.05472	0.05334
Srednje vrijeme zaključivanja	0,10532 s	0,14144 s
Ukupno vrijeme zaključivanja	2,633 s	3,536 s

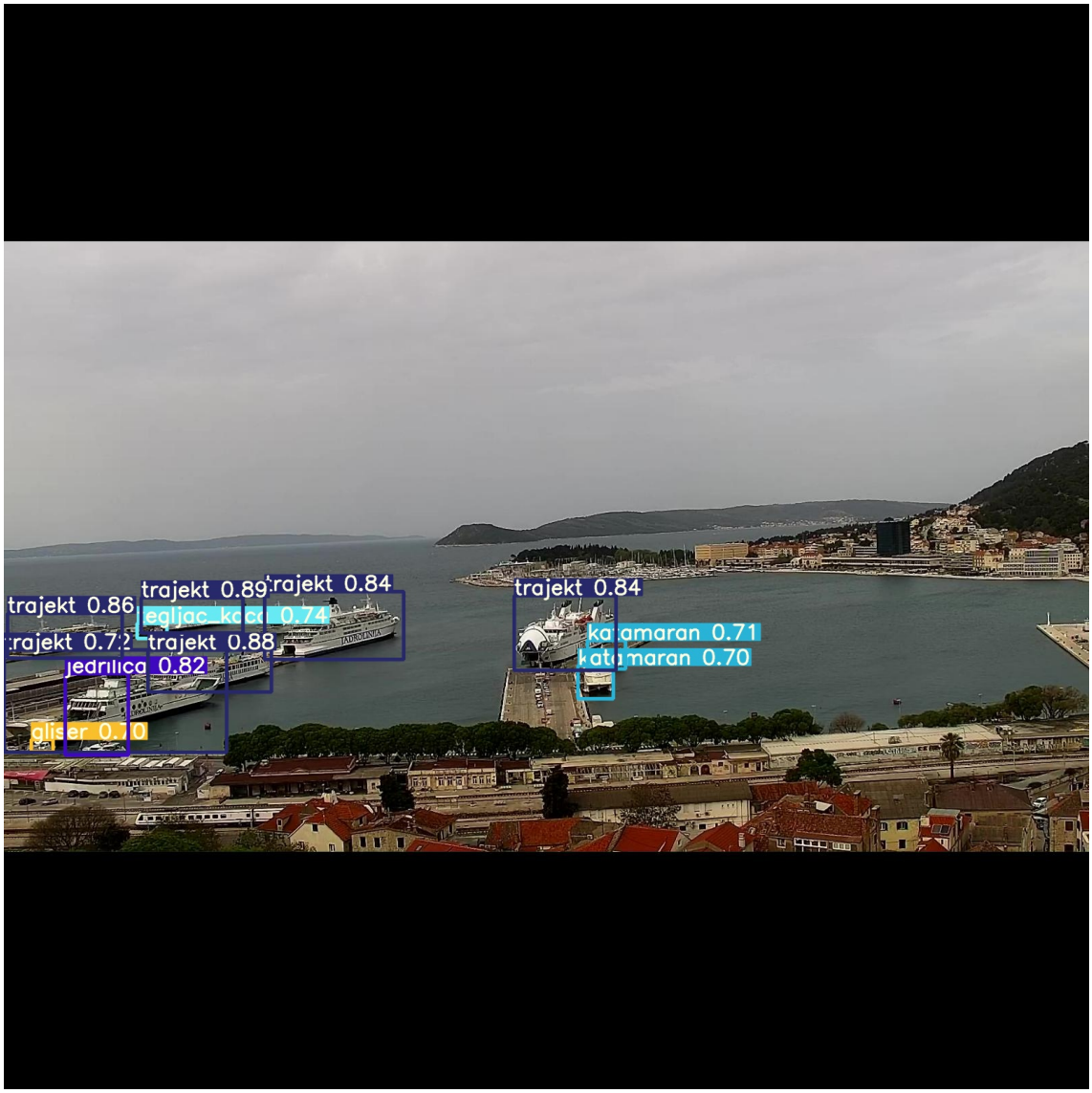
Kao što je iz tablice vidljivo, veći omjer vremena treniranja i broja trenirabilnih parametara kod Small mreže izazvao je podosta bolje rezultate kada ih usporedimo s uloženim vremenom i resursima. Sa više nego dvostruko kraćim vremenom potrebnim za treniranje **Small** mreža dobiva za manje od 7% goru srednju preciznost po klasama pri IOU pragu od 0.5.

Zbog prirode zadatka, praćenje prometa plovila, vrijeme zaključivanja (inferencije) je ključan faktor. Od **Large** mreže smo očekivali podosta veće vrijeme zaključivanja po jednom kadru, što se na kraju ispostavilo pogrešnim. Vrijeme inferencije 25% je bolje kod **Small** mreže, što može (iako ne očekivano značajno) pomoći pri glatkoj obradi ulaznog signala koji bi u ovom slučaju bio video kadar.

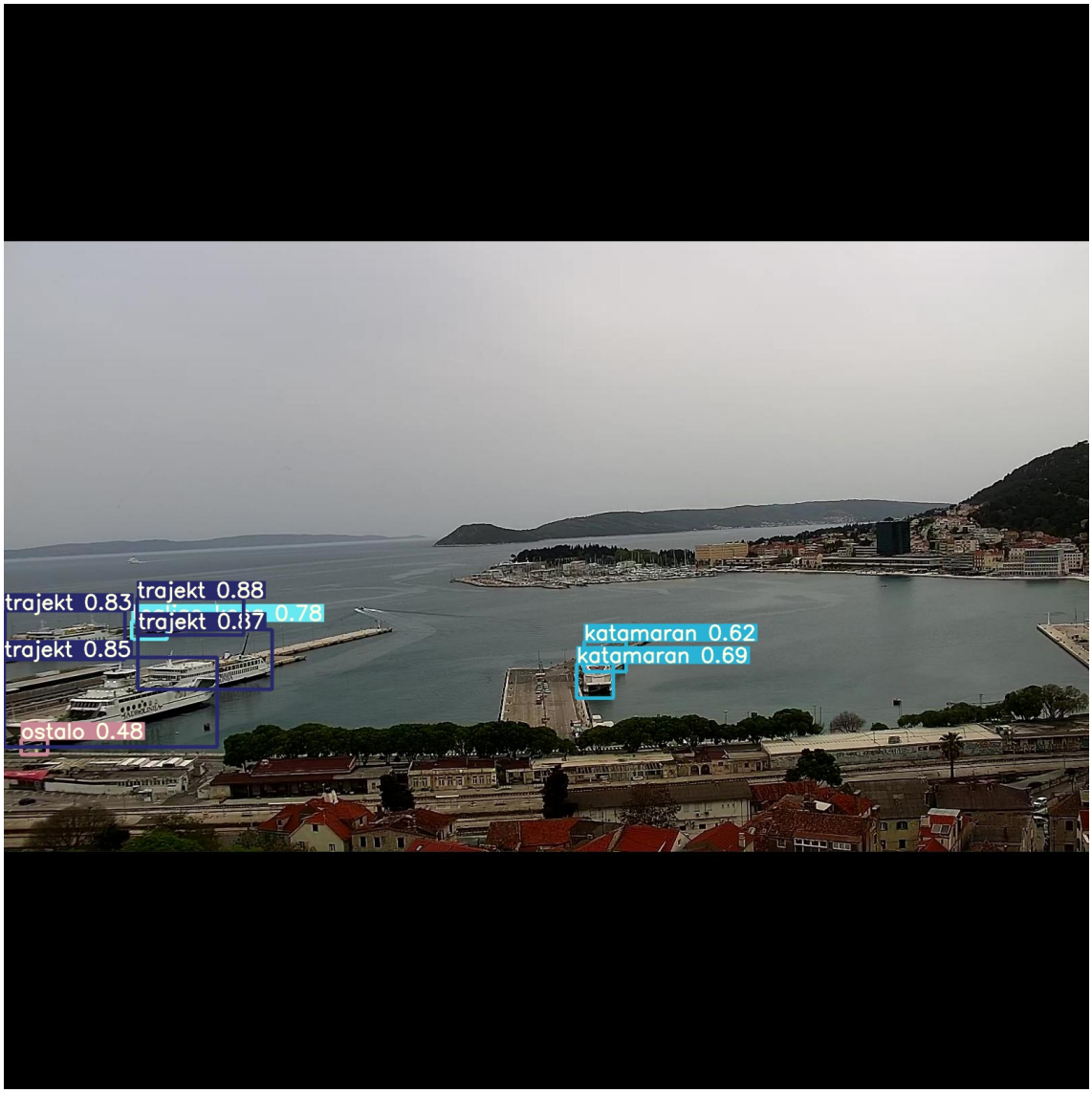


Slika 17: Primjer rada YOLOv5 L modela.

Slika 18: Primjer rada YOLOv5 L modela.



Slika 19: Primjer rada YOLOv5 S modela.



Slika 20: Primjer rada YOLOv5 S modela.

Literatura

[1] Introduction to YOLO Algorithm for Object Detection, Karimi G. , s interneta: https://www.section.io/engineering-education/introduction-to-yolo-algorithm-for-object-detection/#:~:text=YOLO%20is%20an%20algorithm%20that,%2C%20parking%20meters%2C%20and%20animals., zadnji pristup: 1.6.2022.

[2] YOLO: Real-Time Object Detection, Redmon J., s interneta: https://pjreddie.com/darknet/yolo/, zadnji pristup: 1.6.2022.

[3] Custom Object Detection Training using YOLOv5, Rath S., s interneta: https://learnopencv.com/custom-object-detection-training-using-yolov5/, zadnji pristup: 1.6.2022.

[4] YOLO V5—Explained and Demystified, Towards AI, s interneta: https://towardsai.net/p/computer-vision/yolo-v5%E2%80%8A-%E2%80%8Aexplained-and-demystified, zadnji pristup: 1.6.2022.

[5] Auto-Anchor, Ultralytics, s interneta: https://github.com/ultralytics/yolov5/issues/503, zadnji pristup: 1.6.2022.