

## Projektni zadatak 2

### Detekcija objekata pomoću YOLO algoritma

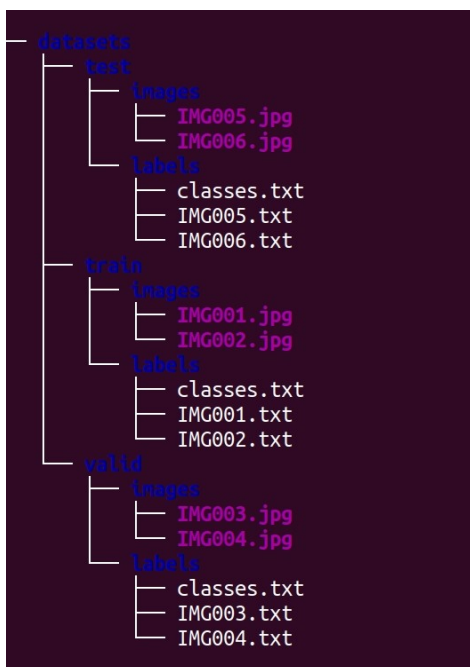
#### UVOD

U prvom delu projektnog zadatka, izvršena je priprema skupa podataka za treniranje YOLO modela za detekciju željenog objekta. U ovom delu zadatka će biti odrađena detekcija. Da bi YOLO model mogao da detektuje željeni objekat, potrebno je istrenirati ga na osnovu pripremljenog skupa podataka.

Vaš zadatak u okviru drugog dela projektnog zadatka je da istrenirate YOLO model na pripremljenom trening skupu. Nakon treniranja, potrebno je odraditi predikciju tj. detekciju objekta iz skupa podataka koji nije korišćen tokom treniranja i obraditi rezultate predikcije.

Ovaj zadatak se sastoji iz 3 celine: treniranje YOLO modela, predikcija i završna obrada (*post-processing*) rezultata predikcije.

**NAPOMENA:** Sve rezultate rada zapisati i komentarisati. Pre početka rada, obavezno utvrditi da vam je skup podataka podeljen na **trening**, **validacioni** i **test** skup, prateći sledeći primer:



Ilustracija 1 Primer podele skupa podataka na trening, validacioni i test skup

## ZADACI

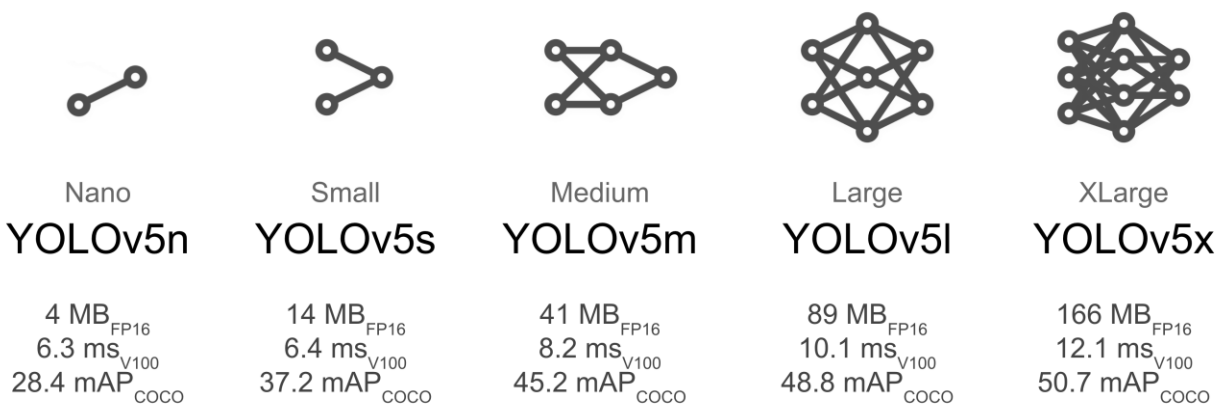
**NAPOMENA: VODITI SE SKRIPTOM flow.sh i dokumentacijom s interneta**

### Zadatak 1

Prateći skriptu flow.sh, pokrenuti treniranje YOLO modela. Da bi treniranje bilo uspešno, potrebno je pripremiti ispravno **data.yaml** datoteku, tako da sadrži putanju do trening i validacionog skupa podataka, broj klasa i nazive klasa koje za koje želite da istrenirate model (*ako imate više klasa, moraju biti navedene u istom redosledu kao u predefined\_classes.txt kada ste radili anotaciju*). Pri treniranju koristiti početni model yolov8s.pt. Ovo je drugi po redu po veličini YOLO model. Za potrebe detekcije objekata, YOLOv8 modeli su unapred istrenirani (eng. *pre-trained*) da prepoznaju 80 klasa objekata (pogledati <https://cocodataset.org/#home>). **Zadatak** modela je **detekcija**, a **mod** je **treniranje**. Argument project će biti putanja foldera gde se čuvaju folderi sa rezultatima treniranja. Argument name će biti naziv foldera na putanji project gde će biti sačuvan rezultat pokretanja komande treniranja. Kao argument **model** postaviti samo **yolov8s.pt**. Ostale argumente poziva komande za treniranje pogledati detaljnije na: <https://docs.ultralytics.com/modes/train/#usage-examples>

Možete istražiti značenje ovih parametara i eksperimentisati s njima: kako utiču na vreme treniranja, preciznost...

Nakon izvršenog treniranja, pogledati sve fajlove dobijene kao rezultate treniranja i pronaći tumačenje rezultata treniranja. Obratiti pažnju na **precision** i **recall**.



Ilustracija 2 Dostupni unapred istrenirani YOLO modeli, po složenosti i veličini modela, na primeru verzije YOLOv5

**NAPOMENA: Da bi se treniranje ispravno izvršilo, potvrdite da su trening i validacioni skup dobro pripremljeni – sve slike imaju labelu, i sve labele imaju naziv isti kao slika (pogledati Ilustracija 1).**

### Zadatak 2

Odraditi predikciju pomoću istreniranog modela. Model će biti skladišten kao rezultat treniranja na putanji gde su rezultati treniranja, u folderu weights. Biće sačuvana dva: best.pt – težine neuronske mreže kada je imala najbolje rezultate nad validacionim skupom podataka i last.pt –

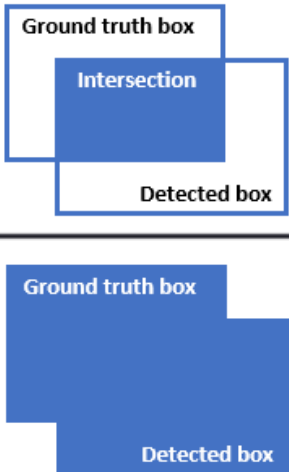
težine neuronske mreže iz poslednje epohe. Uraditi predikciju sa oba nad test skupom i uporediti rezultate. Predikcija se radi tako što se zadatak postavi kao detekcija a mod kao predikcija. Argumente `save` i `save_txt` postaviti na `True`. Ostale argumente za predikciju pogledati na: <https://docs.ultralytics.com/modes/predict/>.

### Zadatak 3

Kao treći zadatak ostavlja se da smislite sami šta biste uradili s rezultatima predikcije, kao korak završne obrade. Neke od ideja su:

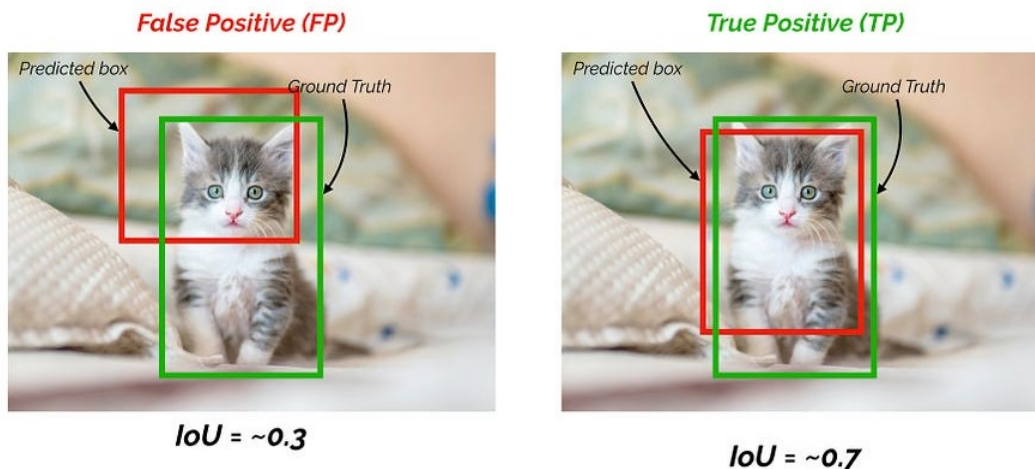
- Segmentacija (rađeno na vežbama) – segmentirati željeni objekat i pozadinu.
- Ukoliko YOLO istrenirani model s početka već sadrži vašu klasu, uraditi poređenje rezultata predikcije s njegovim težinama (koristiti `yolov8s.pt`) sa rezultatima predikcije s vašim težinama (`best.pt` koji dobijete kao rezultat treniranja). Možete izračunati *Intersection over Union*, tj. IoU metriku za oba koliko se poklapaju s originalnom labelom i uporediti.
- Računanje metrika za statistiku o valjanosti modela: preciznost (*precision*) – koliko bounding box-eva kao rezultat predikcije (*detected box*) ima *Intersection over Union* preko 50% sa bounding box-om koji ste ručno obeležili (*ground truth*) u odnosu na broj svih predviđenih bounding box-eva.

Primer: na slici ima samo jedan objekat, a model je predvideo tri objekta; jedan od tih se poklapa 85% sa ručno obeleženom labelom – preciznost je  $1/3=0.33=33\%$ . Izračunati preciznost za svaku sliku posebno, pa zatim za ceo validacioni skup (suma individualnih preciznosti podeljeno sa brojem slika).

$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}} = \frac{\text{Intersection}}{\text{Union}}$$


Ilustracija 3 Intersection over Union metrika

Uporediti preciznost nad test skupom koju ste izračunali sa preciznošću nad validacionim skupom koje dobijete kao rezultat treniranja.



Ilustracija 4 Loše predviđen bounding box (IoU manji od 70%) i dobro predviđen bounding box (IoU oko 70%)

Ukoliko radite analize rezultata, obavezno zapisati rezultate u neki dokument, Excel/Google spreadsheet.

**Windows korisnici:** ukoliko imate problema sa pokretanjem train komande, a greška ima veze sa instalacijom fonta Arial.ttf, preuzmite ga sa: <https://ultralytics.com/assets/Arial.ttf> i iskopirajte ga na putanju: C:\Users\<vaš\_user>\AppData\Roaming\Ultralytics