



UNIVERZITET U NOVOM
SADU
FAKULTET TEHNIČKIH
NAUKA U NOVOM SADU




Alen Mujo

Automatska transformacija snimka utakmice u prikaz iz ptičije perspektive primenom tehnika kompjuterske vizije

Diplomski rad
- Osnovne akademske studije -

Novi Sad, 2021.

	UNIVERZITET U NOVOM SADU FAKULTET TEHNIČKIH NAUKA 21000 NOVI SAD, Trg Dositeja Obradovića 6	Datum:
	ZADATAK ZA IZRADU DIPLOMSKOG (BACHELOR) RADA	List:
		1/1

(Podatke unosi predmetni nastavnik - mentor)

Vrsta studija:	Osnovne akademske studije
Studijski program:	Softversko inženjerstvo i informacione tehnologije
Rukovodilac studijskog programa:	dr Zarić Miroslav

Student:	Alen Mujo	Broj indeksa:	SW 1/2017
Oblast:	Mašinsko učenje		
Mentor:	Dr Jelena Slivka		
NA OSNOVU PODNETE PRIJAVE, PRILOŽENE DOKUMENTACIJE I ODREDBI STATUTA FAKULTETA IZDAJE SE ZADATAK ZA DIPLOMSKI RAD, SA SLEDEĆIM ELEMENTIMA: - problem – tema rada; - način rešavanja problema i način praktične provere rezultata rada, ako je takva provera neophodna; - literatura			

NASLOV DIPLOMSKOG (BACHELOR) RADA:

Automatska transformacija snimka utakmice u prikaz iz ptičije perspektive primenom tehnika kompjuterske vizije.
--

TEKST ZADATKA:

Zadatak ovog rada je specifikacija, implementacija i verifikacija sistema za transformaciju snimka utakmice u prikaz iz ptičije perspektive. Rešenje je implementirano treniranjem postojećih modela za segmentaciju ključnih tačaka ulazne slike i modela za detekciju igrača. Za obučavanje modela korišćen je javno dostupan skup podataka koji se sastoji od slika prenosa utakmica.
--

Rukovodilac studijskog programa:	Mentor rada:

Primerak za: <input type="checkbox"/> - Studenta; <input type="checkbox"/> - Mentora
--

SADRŽAJ

1. Uvod.....	1
2. Prethodna rešenja	3
3. Teorijski pojmovi i definicije.....	6
3.1 Neuronske mreže.....	6
3.2 Konvolucione neuronske mreže.....	7
3.3 Kompjuterska vizija	12
3.4 Detekcija objekata.....	12
3.4.1 YOLO model detekcije objekta.....	14
3.5 Segmentacija slika.....	18
3.5.1 U-Net arhitektura	20
3.6 Homografija	22
4. Specifikacija i implementacija rešenja	24
4.1 Skup podataka	24
4.2 Specifikacija rešenja	25
4.2.1 Segmentacija i detekcija ključnih tačaka.....	27
4.2.2 Detekcija igrača	29
4.2.3 Klasterizacija igrača	30
5. Evaluacija rešenja i rezultati	32
5.1 Evaluacija modela segmentacije	32
5.2 Evaluacija modela detekcije igrača.....	33
6. Zaključak.....	35
7. Literatura.....	36
KLJUČNA DOKUMENTACIJSKA INFORMACIJA.....	39
KEY WORDS DOCUMENTATION.....	40

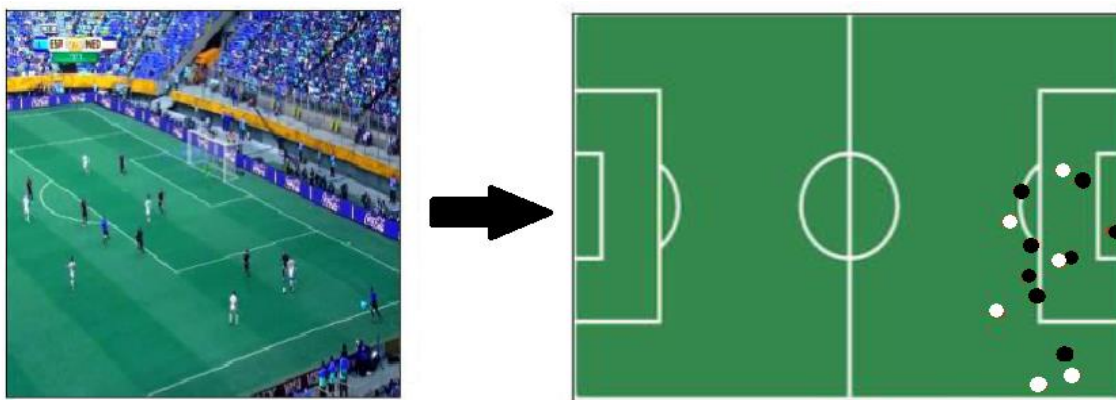
1. Uvod

Tokom protekle decenije, povećanjem dostupnosti i količine podataka, su oblasti *Data Science* (DS) i Mašinsko učenje (*engl. ML - Machine Learning*) postali važan deo svakog velikog sportskog tima. U radu [47] su autori izvršili pregled oblasti i pojasnili na koje su sve načine DS/ML doprineli fudbalskoj analitici, koje probleme rešavaju ali i koji su sve problemi i dalje nerešivi. Podaci imaju veliku ulogu u donošenju odluka o izboru igrača, evaluaciji performansi samih igrača i timova (distanca koju je igrač prešao, brzina kojom se kretao, pozicioniranje igrača u određenim situacijama), taktičkoj analizi protivničkih timova i dobavljanju individualne statistike igrača. Postoje dva tipa podataka koji su najzastupljeniji u analizi:

1. *Event data* - predstavlja podatke koji uglavnom uključuju sve događaje koji su se desili i vezani su za fudbalsku loptu. Na primer: naziv ili broj igrača koji je imao kontakt sa loptom i njegova pozicija, vremenska oznaka kada se događaj dogodio, tip događaja (dodavanje, korner, slobodni udarac, penal, šut u okvir gola, ...), ishod događaja (na primer: promašaj ili uspešno dodavanje). Ovaj tip podataka je moguće ručno skupljati analizom i pregledanjem snimka same utakmice. Nedostatak ovih podataka je taj što izostavljaju sve događaje u kojima učestvuje preostalih 21 igrača koji nemaju posed lopte a koji predstavljaju 97% događaja koji se dogode u toku jedne utakmice.

2. *Tracking data* - predstavlja podatke svakog igrača na terenu u određenim vremenskim intervalima. Na primer: koordinate svakog igrača se beleže na svakih 0.2 sekunde. Precizno skupljanje ove vrste podataka je gotovo nemoguće izvršiti ručno. Jedan od najzastupljenijih načina na koji se dolazi ove vrste podataka je uz korišćenje uređaja i senzora koji prate pozicije igrača ili korišćenjem nekoliko kamera koje snimaju igru iz različitih uglova. Ovakav način prikupljanja podataka, iako dosta pouzdan i precizan, predstavlja problem jer zahteva instaliranje i održavanje skupe opreme koja nije uvek dostupna manjim timovima.

U ovom radu je predložena implementacija rešenja za prikupljanje *Tracking* podataka iz slika koje su izdvojene iz video snimka prenosa fudbalske utakmice. Skup podataka je predstavljen u [1]. Ulaz u sistem predstavlja slika fudbalskog terena iz snimka prenosa utakmice a izlaz predstavlja prikaz pozicija igrača na terenu (Slika 1).



Slika 1 - Prikaz ulaza u sistem (levo) i izlaza (desno).

Rešenje je implementirano kroz tri odvojena modula:

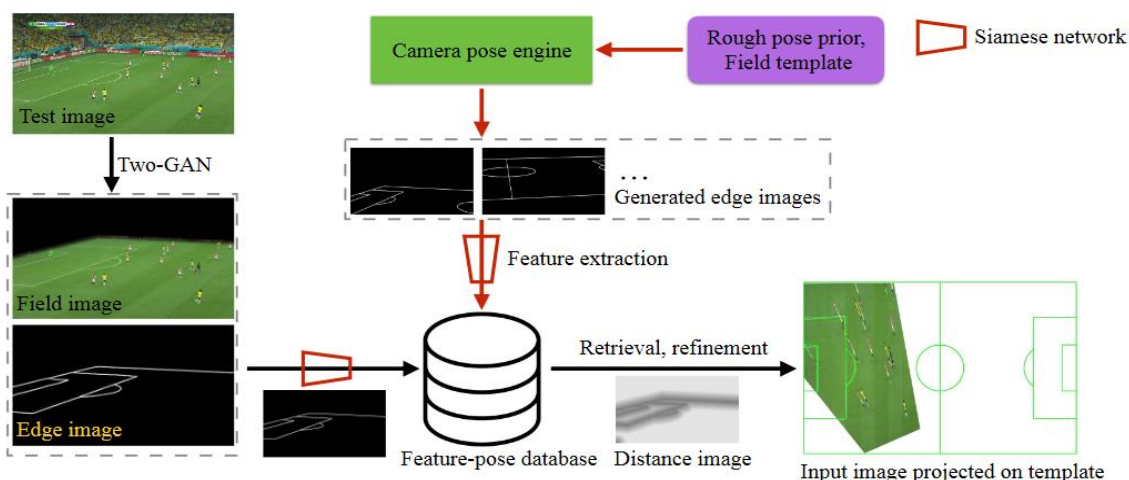
- Prvi modul radi segmentaciju ključnih tačaka ulazne slike (preseki linija fudbalskog terena) i računanje homografije između ulazne slike i njene reprezentacije u dvodimenzionalnom prostoru. Korišćen je U-Net model za segmentaciju ključnih tačaka jer se smatra dobrim modelom za male skupove podataka [30]. Model je evaluiran na testnom skupu i ostvario rezultat od 83% IoU (*engl. Intersection over Union*).
- Drugi modul se bavi detekcijom igrača na ulaznoj slici i transformacijom njihovih koordinata u dvodimenzionalni prostor korišćenjem homografije izračunate u prethodnom koraku. Korišćen je YOLO model zbog svoje preciznosti ali i brzine detekcije igrača [26]. Model je postigao rezultat od 0.98 mAP (*engl. mean Average Precision*).
- Na kraju je upotrebom *k-means* algoritma izvršena klasterizacija vrednosti piksela detektovanih igrača kako bi se odredila boja igrača prilikom njegovog prikaza na izlaznoj slici.

U narednom poglavlju je dat pregled postojećih rešenja koja su koristila slični skup podataka kao i ovaj rad. U trećem poglavlju su predstavljeni teorijski pojmovi i opisi korišćenih modela koji su važni za razumevanje predstavljenog rešenja. Specifikacija rešenja segmentacije, detekcije igrača i klasterizacije piksela je objašnjena u četvrtom poglavlju. U petom poglavlju je opisan način evaluacije korišćenih modela i predstavljeni su dobijeni rezultati dok poslednje poglavlje predstavlja sumarizaciju rada.

2. Prethodna rešenja

Prethodna rešenja opisana u ovom poglavlju koriste slični skup podataka koji se koristi i u ovom radu, *world cup 2014* [1], sa ciljem da na osnovu ulazne slike, koja predstavlja prikaz fudbalskog terena sa igračima, prediktuju homografiju i generišu izlaznu sliku u 2D prostoru. Skup podataka se sastoji od 209 trening i 186 test slika dimenzija 320 x 320 x 3.

U radu [2] su autori primenili dve GAN (*engl. Generative Adversarial Networks*) mreže za obradu ulazne slike (Slika 2). Prva GAN mreža uklanja sa ulazne slike pozadinu, publiku i sve što nije deo fudbalskog terena. Kao rezultat je dobijena slika na kojoj je izdvojena regija fudbalskog terena. Dobijena slika je ulaz u drugu GAN mrežu koja za zadatak ima da detektuje linije fudbalskog terena i generiše sliku na kojoj će samo te linije biti vidljive a svi ostali objekti kao što su igrači i lopta biti uklonjeni. Nakon toga se koristi prethodno trenirana *siamese* mreža za ekstrakciju osobina iz slike i kao izlaz daje vektor osobina (*engl. feature vector*). Dobijen vektor se dalje koristi za pronalaženje slične slike iz baze vektora osobina metodom najbližih suseda (*engl. kNN - k-nearest neighbours*) čime je moguće preuzeti homografiju najbližnje slike. Dobijena homografija se dalje koristi za transformaciju ulazne slike u 2D projekciju.

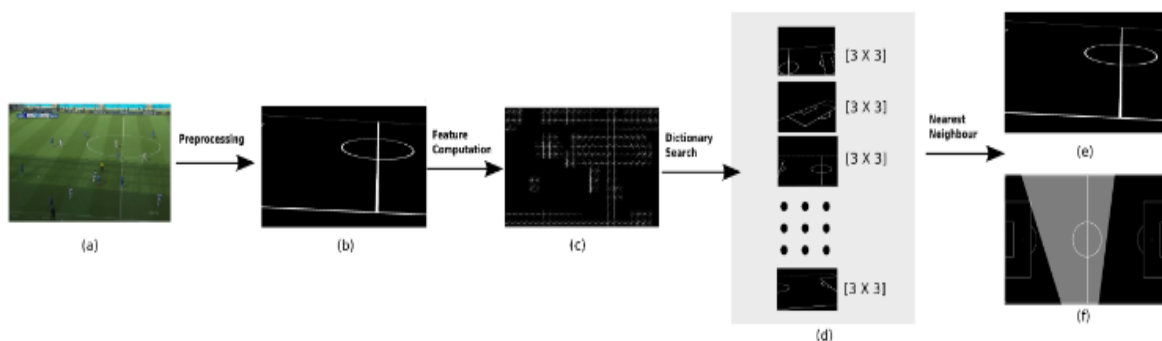


Slika 2 - Dve GAN mreže vrše obradu ulazne slike i kreiraju sliku ivica (*engl. edge image*). *Siamese* mreža vrši izdvajanje osobina i dobijeni vektor se metodom najbližih suseda spaja sa najbližim vektorom u bazi. Pronađena homografija se koristi za transformaciju ulazne slike. Preuzeto iz [2].

Autori su za evaluaciju koristili metriku IoU (*engl. Intersection over Union*) i ostvarili rezultat od 87%. Njihov metod je takođe pokazao dobre rezultate na skupu podataka koji se sastojao od slika odbojkaške utakmice.

U radu [3] su autori na ulaznu sliku primenili *Pix2Pix* [4] generativni model kako bi generisali sliku kod koje je uklonjeno sve osim linija fudbalskog terena (Slika 3). Na generisanu sliku se primenjuje HOG (*engl. Histogram of Oriented Gradients*) deskriptor za ekstrakciju osobina i kNN za pronalaženje najbliže slike u bazi koja se sastoji od slika i odgovarajućih homografija. Dobijena homografija se primenjuje za projekciju ulazne slike u *top down view* koji predstavlja izlaz. Za potrebe detekcije igrača se ulazna slika obrađuje uklanjanjem zelenih piksela čime se dobija slika na kojoj su samo igrači, linije i delovi pozadine. Vrš se detekcija kontura i smatra se da svaka kontura koja ima visinu 1.5 puta veću od širine predstavlja detektovanog igrača. Nad selektovanom regijom se primenjuje klasterizacija piksela kako bi se odredila

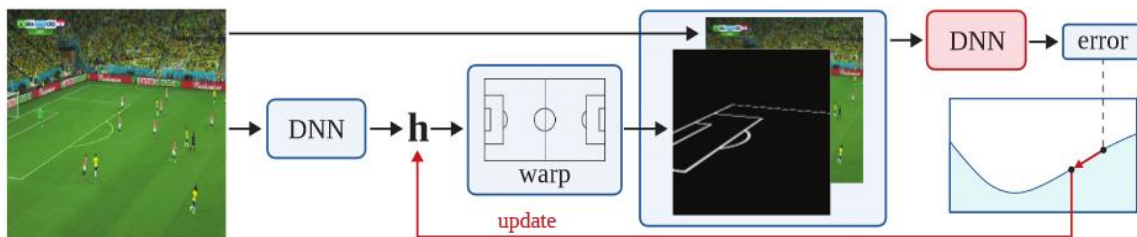
boja dresa igrača koja služi za prikaz. Primenom homografije na koordinate detektovanih igrača se vrši projekcija igrača na izlaznu sliku.



Slika 3 - Na ulaznu sliku (a) se primenjuje generativni model *Pix2Pix* koji generiše sliku (b). Na sliku (b) se primenjuje *HOG* deskriptor za ekstrakciju vektora osobina. Metodom najbližih suseda se u bazi pronalazi najslbližji vektor i izdvaja homografija koja se koristi za transformaciju ulazne slike. Preuzeto iz [3].

Za evaluaciju je korišćena IoU metrika i ostvaren je rezultat od 83%. Autori ističu da je kod detekcije igrača moguće izgubiti igrače u susednim frejmovima i predlažu korišćenje metoda za procenu kretanja igrača. Takođe je potrebno povećati bazu za pretragu. Autori su svoj metod primenili i na drugi sport (hokej) i ostvarili dobre rezultate.

U radu [5] se na ulaznu sliku primenjuje duboka neuronska mreža (*DNN - engl. Deep Neural Network*) koja kao rezultat daje procenu homografije (*h*) (Slika 4). Na ulaznu sliku se primenjuje procenjena homografija i transformisana slika se u narednom koraku šalje u drugu duboku neuronsku mrežu koja poredi transformisanu sliku sa *ground truth* slikom i računa grešku. Glavna ideja je diferencijacijom odrediti kako pomeriti početnu predikciju (*h*) kako bi smanjili grešku koju daje druga mreža i time poboljšati procenu (*h*). Ovo se ponavlja do konvergencije. Mana ovog postupka je što metoda zavisi od prvobitne procene homografije (*h*) koja može dovesti do konvergencije ka lokalnom optimumu. Brzina takođe predstavlja problem, jer se prilikom svake iteracije, slika mora transformisati pre ulaska u drugu *DNN* mrežu i procene greške. Procenjena homografija se na kraju primenjuje na ulaznu sliku za transformaciju u 2D prostor i predstavlja izlaz. Za evaluaciju je korišćena IoU metrika i ostvaren je rezultat od 83%.



Slika 4 - Ulazna slika se prosleđuje prvoj dubokoj neuronskoj mreži koja je trenirana da kao izlaz da predikciju homografije - *h*. Slika se transformiše upotrebom homografije *h* i prosleđuje u drugu duboku neuronsku mrežu koja radi procenu greške poređenjem sa *ground truth* slikom. Ideja je odrediti kako promeniti inicijalnu procenu *h* kako bi smanjili grešku. Ovi koraci se ponavljaju do konvergencije. Preuzeto iz [5].

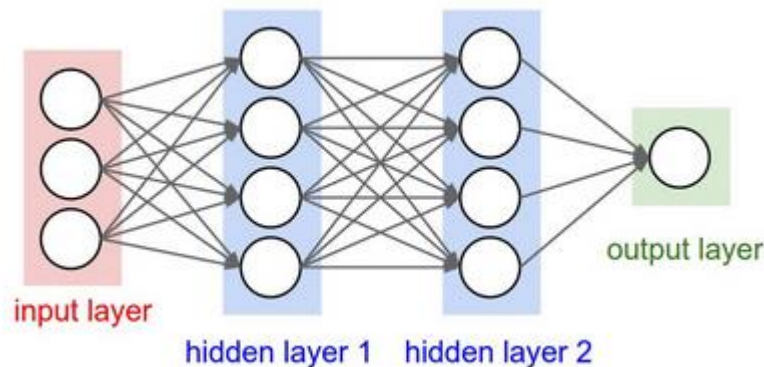
Rešenje opisano u ovom radu, za razliku od pomenutih metoda, koristi U-Net model segmentacije za detekciju ključnih tačaka na osnovu kojih se računa homografija. Izlaz je, kao i u radu [3], projekcija detektovanih igrača procenjenom homografijom uz razliku da je u ovom radu za detekciju igrača korišćena konvoluciona neuronska mreža.

3. Teorijski pojmovi i definicije

U ovom poglavlju su predstavljene teorijski pojmovi važni za razumevanje korišćenih metoda u radu. U poglavljima 3.1 i 3.2 su objašnjene neuronske mreže i konvolucione neuronske mreže. Nakon toga su u poglavlju 3.3 definisani standardni problemi kompjuterske vizije. U ovom radu je problem detekcije igrača rešavan pomoću YOLO algoritma koji je pojašnjen u poglavlju 3.4. U poglavlju 3.5 je definisan pojam segmentacije slike i objašnjena je arhitektura modela (U-Net) koji je korišćen za izdvajanje ključnih tačaka ulazne slike. Problem homografije koja se koristi za mapiranje slike na 2D model je objašnjen u poglavlju 3.6.

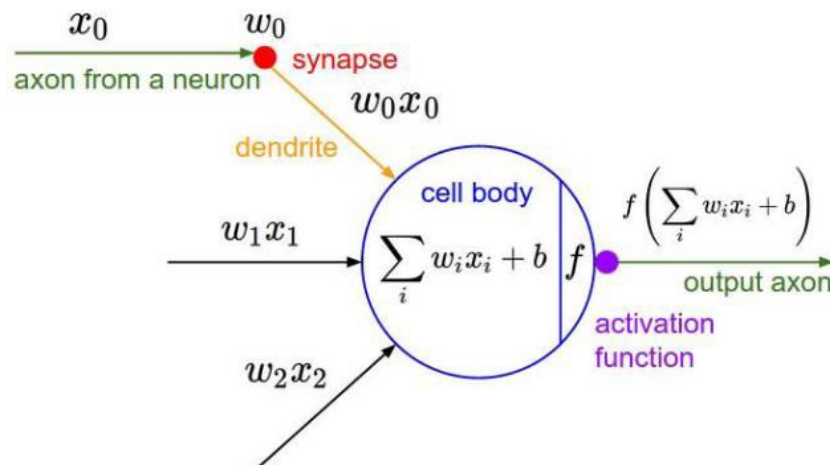
3.1 Neuronske mreže

Neuronske mreže predstavljaju kolekciju neurona povezanih u aciklični graf, organizovanih u slojeve, u kojima su izlazi nekih neurona iz jednog sloja ulazi za druge neurone u drugom sloju (Slika 5). Prvi sloj prihvata ulazne vrednosti se naziva ulazni sloj. Ulazna vrednost koja je predstavljena vektorom se transformiše prolaskom kroz niz skrivenih slojeva koji se nalaze između ulaznog i izlaznog sloja i sastoje se od nekoliko neurona koji su potpuno povezani sa svakim neuronom iz prethodnog sloja. Za neurone u jednom sloju važi da su potpuno nezavisni i nemaju međusobne konekcije. Poslednji sloj se zove izlazni sloj i vrednosti njegovih neurona predstavljaju verovatnoće svake klase (*engl. class scores*) u slučaju klasifikacije ili realne vrednosti u slučaju regresije.



Slika 5 - Troslojna neuronska mreža sa tri ulaza, dva skrivena sloja od po 4 neurona i jednim izlaznim slojem. Preuzeto iz [33].

Svaki neuron računa sumu proizvoda svojih ulaznih vrednosti i težina ($w_0 \cdot x_0 + w_1 \cdot x_1 + w_2 \cdot x_2$). Na dobijeni zbir se dodaje broj koji predstavlja *bias*. Pre slanja dobijene vrednosti neuronu u narednom sloju, primenjuje se nelinearna aktivaciona funkcija koja proizvodi broj koji predstavlja izlaz neurona (Slika 6). Uloga aktivacione funkcije je da u model uvede nelinearnost.



Slika 6 - Prikaz operacija u jednom neuronu. Prezeto iz [33].

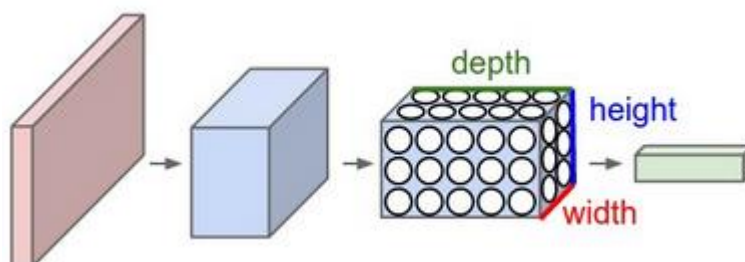
Učenje u neuronskoj mreži se svodi na podešavanje težina (w_0, w_1, \dots, w_n) i biasa (b_0, b_1, \dots, b_n) u svakom od neurona. Prilikom treniranja neuronske mreže, za svake ulazne vrednosti su vezane i vrednosti koje predstavljaju tačan rezultat neuronske mreže. Poređenjem tačne vrednosti i vrednosti koju daje neuronska mreža se dobija procenjena greška. Funkcija koja određuje razliku tačne i predviđene vrednosti se zove funkcija greške (*engl. cost function*). Ideja je smanjiti vrednost funkcije greške što je više moguće. Algoritam za obučavanje se zove *backpropagation* i računa gradijent funkcije greške i govori nam koliko treba promeniti težine svih neurona kako bi se vrednost funkcije greške minimizovala.

3.2 Konvolucione neuronske mreže

Konvolucione neuronske mreže (*engl. Convolutional Neural Network - CNN*) su vrsta neuronskih mreža najčešće korišćena kada ulaz predstavlja slika. Zovu se konvolucione neuronske mreže jer koriste matematičku operaciju konvolucija u bar jednom svom sloju. CNN predstavljaju *state-of-the-art* rešenje za zadatak klasifikacije slika [6]. Prvi put su predložene 1988 u radu [7] i nisu imale široku upotrebu zbog hardverskih ograničenja prilikom treniranja modela. 1990-ih, LeCun je primenio *gradient-based* algoritme učenja na CNN i postigao dobre rezultate za problem klasifikacije ručno pisanih brojeva [8]. Nakon njega, mnogi istraživači su dalje unapredili CNN i došli do *state-of-the-art* rezultata za probleme kompjuterske vizije.

Optimizovanost u procesiranju slika i ekstrakciji osobina predstavljaju prednost CNN u odnosu na neuronske mreže. Ukoliko se neuronskom mrežom radi klasifikacija slike dimenzije $32 \times 32 \times 3$ (32 širina, 32 visina, 3 kanala boje - (R)ed, (G)reen, (B)lue), onda će jedan neuron u skrivenom sloju neuronske mreže imati $32 \times 32 \times 3 = 3072$ težina. U zavisnosti od konačnog broja neurona i same strukture neuronske mreže, ovo može dovesti do velikog broja parametara koje je potrebno trenirati što znatno utiče na brzinu treniranja i samu praktičnu upotrebljivost modela. Još jedna mana primene klasičnih neuronskih mreža na analizu slike je ta što se predstavljanjem slike kao jednodimenzionalnog vektora gube prostorne osobine piksela i međusobni odnosi između susednih piksela kao i relacije sa drugim objektima na slici. Prednost konvolucionih neuronskih mreža su deljenje parametara unutar mreže i čuvanje prostornih osobina.

Neuroni u slojevima konvolucione neuronske mreže su organizovani u tri dimenzije, širina, visina i dubina, gde svaki sloj transformiše trodimenzionalni ulaz u trodimenzionalni izlaz. Za razliku od slojeva običnih neuronskih mreža, neuroni u konvolucionim neuronskim mrežama su povezani sa malom regijom iz prethodnog sloja (Slika 7).



Slika 7 - Primer organizacije jedne konvolucione neuronske mreže (CNN). Preuzeto iz [33].

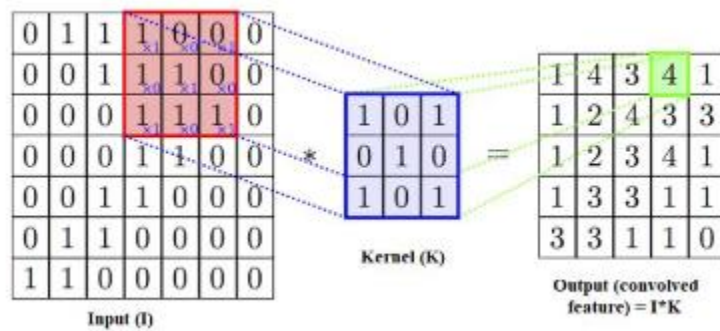
CNN se obično sastoji od jednog ili više blokova konvolucionih i *pooling* slojeva koji su praćeni jednim ili više potpuno povezanim slojevima i izlaznim slojem. Kombinacija ovih slojeva čini konvolucionu neuronsku mrežu (Slika 8).



Slika 8 - Slojevi koji čine konvolucionu neuronsku mrežu (CNN). Preuzeto iz [34].

Konvolucionni sloj

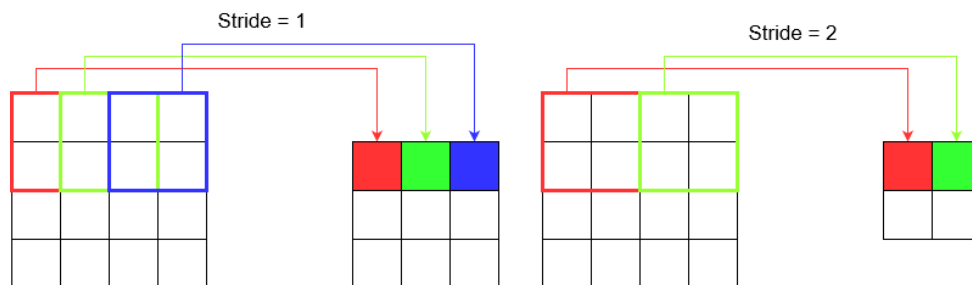
Konvolucionni sloj predstavlja osnovu svake konvolucione neuronske mreže. Primenom operacije konvolucije jednog ili više filtera/kernela (dvodimenzionalnog niza) na ulaz, proizvodi se dvodimenzionalni izlaz koji predstavlja niz osobina (*engl. feature map*). Kernel se primenjuje na ulaz tako što se pomera po širini i visini dvodimenzionalnog ulaza, vršeći operaciju množenja (*dot product*) težina sa vrednostima ulaza. Nakon sumiranja pomnoženih vrednosti, rezultat je jedan broj (Slika 9). Nakon što se filter primeni na svim delovima slike, rezultat predstavlja dvodimenzionalni niz osobina. Korišćenje filtera čije su dimenzije manje od dimenzija ulaza nam omogućava da se težine jednog filtera primene na više delova ulaza, što dovodi do deljenja parametara težina filtera. Ideja iza toga je da, ukoliko je filter takav da detektuje određenu osobinu ulaza (npr. vertikalne linije), onda će primenom tog filtera na više delova istog ulaza omogućiti pronalazak iste osobine na više mesta ulazne slike. Vrednosti težina filtera se kao i kod običnih neuronskih mreža uče algoritmom *backpropagation*.



Slika 9 - Kernel/filter dimenzije 3 x 3 se primenjuje na celoj slici množenjem vrednosti filtera sa odgovarajućim vrednostima ulaza. Zbir proizvoda se smešta na odgovarajuću poziciju u izlazu. Preuzeto iz [34].

Dimenzije izlaza konvolucionog sloja određene su sa 4 hiperparametra:

- K - Predstavlja broj filtera. Uloga filtera je da u procesu treniranja nauči da prepoznaje određene osobine na slici. Veći broj filtera omogućuje učenje većeg broja osobina, npr. ivica, horizontalnih ili vertikalnih linija, oblika, boja.
- F - Predstavlja dimenzije filtera.
- S - *Stride* predstavlja veličinu koraka kojom se filter pomera u svakoj iteraciji. Ako je vrednost ovog parametra 1, to znači da će se filter u svakom koraku pomerati za jedan piksel u desno do kraja reda, nakon čega će se pomeriti na početak reda i jedan piksel na dole (Slika 10).



Slika 10 - Primer pomeranja kernela za vrednost parametra *stride* = 1 i 2. Preuzeto iz [34].

- P - *Padding* predstavlja način za kontrolisanje dimenzije izlaza nakon primene kernela dodavanjem rubnih piksela čija je vrednost 0 (Slika 11).

0	0	0	0	0	0
0	35	19	25	6	0
0	13	22	16	53	0
0	4	3	7	10	0
0	9	8	1	3	0
0	0	0	0	0	0

Slika 11 - Primer *padding*-a vrednosti 1. Preuzeto iz [35].

Znajući hiperparametre, možemo izračunati dimenzije izlaza nakon primene filtera: neka su W_1 , H_1 , D_1 visina, širina i dubina dvodimenzionalnog ulaza. Vrednosti W_2 , H_2 i D_2 (visina, širina i dubina) dvodimenzionalnog izlaza se definišu kao:

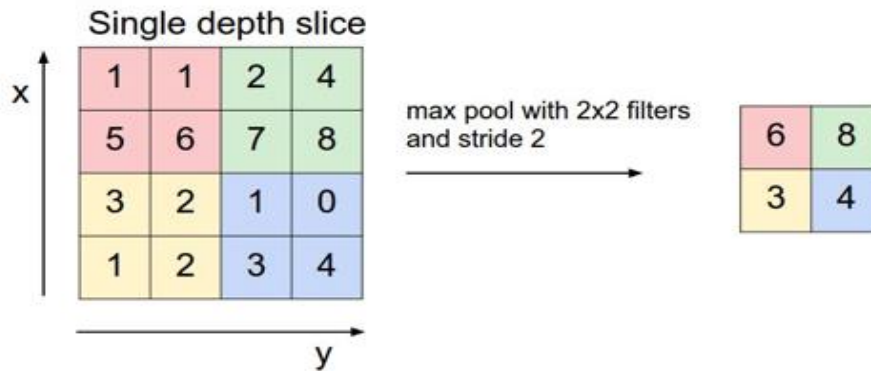
$$W_2 = \frac{W_1 - F + 2P}{S} + 1$$

$$H_2 = \frac{H_1 - F + 2P}{S} + 1$$

$$D_2 = K.$$

Pooling sloj

Između konvolucionih slojeva se nalazi *pooling* sloj čija uloga je da smanji dimenzionalnost izlaza konvolucionog sloja kako bi se smanjio broj parametara koje je potrebno naučiti treniranjem (Slika 13). Time se smanjuje vreme potrebno za treniranje mreže. *Maxpool* je primer jednog *pooling* sloja koji na primeru sa slike 12 uzima maksimalnu vrednost iz regije slike dimenzije 2 x 2 i smešta je u izlaznu matricu dimenzije 2 x 2.



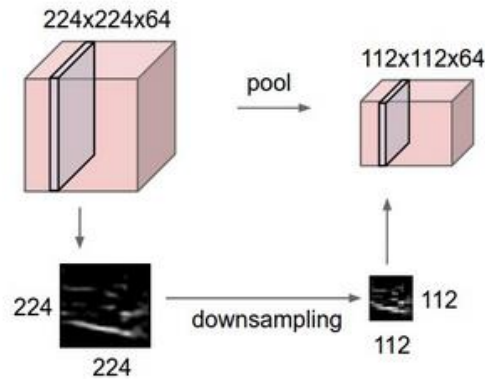
Slika 12 - *Maxpooling* - maksimalna vrednost iz regija dimenzije 2 x 2 se smešta u izlaz. Preuzeto iz [34].

Dimenzije izlaza nakon *pooling* sloja se računaju po formuli:

$$W_2 = \frac{W_1 - F}{S} + 1$$

$$H_2 = \frac{H_1 - F}{S} + 1$$

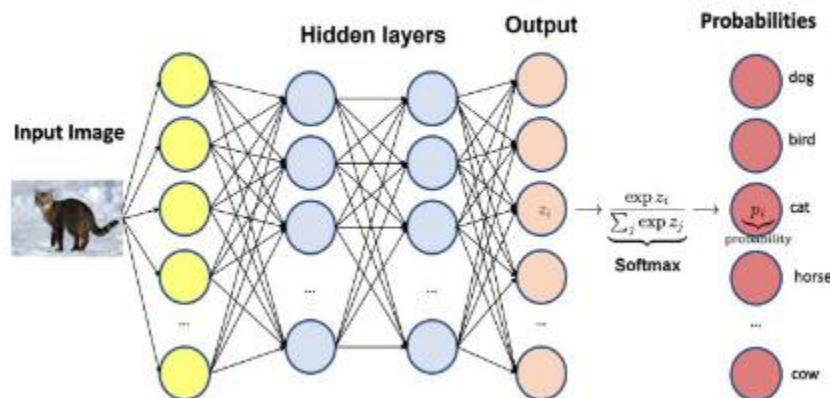
$$D_2 = D_1.$$



Slika 13 - *Pooling* sloj smanjuje širinu i visinu za dva puta. *Pooling* sloj nema parametre koje je treniranjem potrebno naučiti. Preuzeto iz [33].

Potpuno povezani sloj

Poslednji slojevi u CNN su obično potpuno povezani slojevi. Ovaj sloj nastaje pretvaranjem dvodimenzionalnog izlaza iz prethodnog konvolucionog sloja u jednodimenzionalni vektor. Na primer, izlaz $32 \times 32 \times 3$ će biti smešten u vektor veličine 3072. Ovaj sloj se dalje povezuje sa poslednjim slojem, obično *softmax* slojem u slučaju klasifikacije slika. *Softmax* sloj ima zadatak da numeričke vrednosti iz prethodnog sloja pretvori u verovatnoće, tako što uzima eksponent svakog broja i normalizuje svaki broj sumom svih eksponenata, tako da zbir čitavog vektora daje 1. Na primer, ako želimo da klasifikujemo 4 objekta (mačka, pas, čovek, kuća), naš izlazni vektor bi imao 4 broja (0.1, 0.02, 0.04, 0.84) koji predstavljaju verovatnoće. Zaključak bi bio da je verovatnoća da se na slici nalazi kuća 84% a da je verovatnoća da se na slici nalazi pas samo 2% (Slika 14).

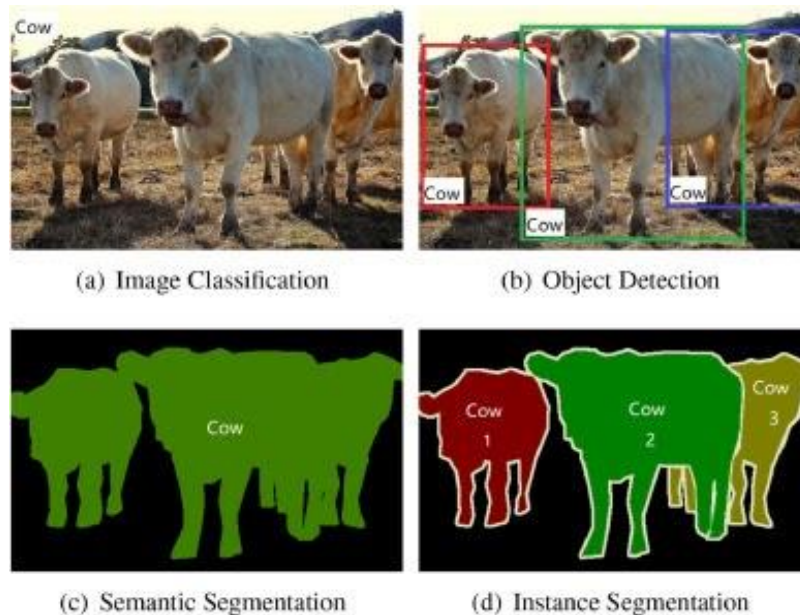


Slika 14 - Primer klasifikacije slike i *softmax* sloja. Preuzeto iz [34].

3.3 Kompjuterska vizija

Kompjuterska vizija je oblast kompjuterske nauke čiji je cilj da računarima obezbedi mogućnost razumevanja slika ili videa. Neke od aplikacija kompjuterske vizije su autonomna vozila [9], sistemi za prepoznavanje lica [10], prepoznavanje pokreta [11], detekcija i prepoznavanje objekata [12], robotika [13] i mnoge druge. Zadatak prepoznavanja i detekcije se može podeliti na 4 pod oblasti:

- Klasifikacija slika - ulaz predstavlja slika a izlaz je klasa/kategorija kojoj slika pripada (Slika 15.a.).
- Detekcija objekta - zadatak je na ulaznoj slici detektovati i označiti objekat (lokalizacija objekta) i svakom detektovanom objektu dodeliti klasu/kategoriju (klasifikacija objekta) (Slika 15.b.).
- Semantička segmentacija - dodeljivanje labele koja predstavlja klasu/kategoriju svakom pikselu na slici, bez razlikovanja pojedinačnih instanci objekata (Slika 15.c.).
- *Instance* segmentacija - razlika u odnosu na semantičku segmentaciju je u tome što se pravi razlika između instanci objekata iste klase/kategorije (Slika 15.d.).



Slika 15 - Zadaci kompjuterske vizije. (a) klasifikacija slike, (b) detekcija i klasifikacija objekta, (c) semantička segmentacija, (d) *instance* segmentacija. Preuzeto iz [12].

3.4 Detekcija objekata

Pre široke upotrebe *deep learning* metoda, zadatak detekcije objekata se mogao podeliti u tri koraka [14]:

- Generisanje regija od interesa - cilj je skenirati celu sliku tehnikom klizajućeg prozora kako bi se pronašle regije od interesa koje imaju veliku verovatnoću da sadrže objekat.
- Ekstrakcija osobina (*engl. feature extraction*) - izdvajanje vektora osobina koji predstavlja semantičke informacije svake regije od interesa na slici.
- Klasifikacija regije - dodeljivanje labele koja predstavlja klasu/kategoriju koja najbolje opisuje koji objekat se nalazi na regiji izdvojenoj u prethodnom koraku.

Ovaj način detekcije i klasifikacije objekata je imao nekoliko nedostataka:

- generisanje i pronalazak regija od interesa je spor proces jer zbog različitih dimenzija mogućih objekata na slici zahteva menjanje dimenzija klizajućeg prozora pri svakoj iteraciji (Slika 16),
- klasifikator se mora primenjivati na svakoj poziciji za svaku dimenziju klizajućeg prozora,
- dimenzije samog prozora se određuju ručno,
- veliki broj izdvojenih regija od interesa su pogrešne i ne sadrže objekte od interesa,
- korak klasifikacije regije se mora trenirati posebno na skupu podataka koji sadrži objekte od interesa.



Slika 16 - Detekcija automobila tehnikom klizajućeg prozora zahteva nekoliko iteracija različitim dimenzijama klizajućeg prozora zbog dimenzija objekata koji se detektuje na slici. Preuzeto iz [14].

Trenutni *state-of-the-art* modeli za detekciju i klasifikaciju objekata su zasnovani na *deep learning* metodama koje su uspele da grešku klasifikacije spuste ispod 4% na *ImageNet* skupu podataka [6] (Slika 17).



Slika 17 - Poređenje rezultata na *ImageNet* skupu podataka između tradicionalnih i *deep learning* metoda. Preuzeto iz [14].

Mogu se podeliti na dve grupe detektora:

1. *two-stage* detektori ili *region proposal-based* metode
2. *one-stage* detektori ili *regression-based* metode

Detektori sa dve faze zadatak detekcije i klasifikacije dele na dva koraka. U prvom koraku generišu skup predloga regija identifikujući delove slika koje potencijalno sadrže objekte. U drugoj fazi *deep learning* model dodeljuje labele svakoj od regija predloženih u prethodnoj fazi. Detektori sa dve faze obično ostvaruju bolje rezultate na mnogim poznatim skupovima podataka ali je njihova mana brzina. U ovu

grupu spadaju R-CNN [15], SPP-net [16], Fast R-CNN [17], Faster R-CNN [18], R-FCN [19] i Mask R-CNN [20].

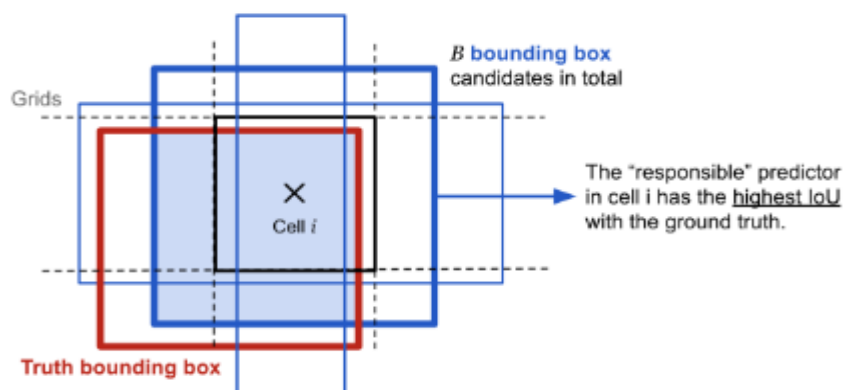
Detektori sa jednom fazom nemaju posebnu fazu za predlog regija. Umesto toga, oni svaku poziciju posmatraju kao potencijalni objekat i pokušavaju da svakoj regiji od interesa dodele labelu. Zbog ovoga su dosta efikasniji u pogledu vremena za detekciju i klasifikaciju i poželjni su za detekciju u realnom vremenu. U ovu grupu spadaju MultiBox [21], AttentionNet [22], G-CNN [23], YOLO [24], SSD [25] i YOLOv2 [26].

U ovom radu je korišćen YOLO model za detekciju igrača zbog svoje preciznosti ali i brzine treniranja i detekcije igrača.

3.4.1 YOLO model detekcije objekta

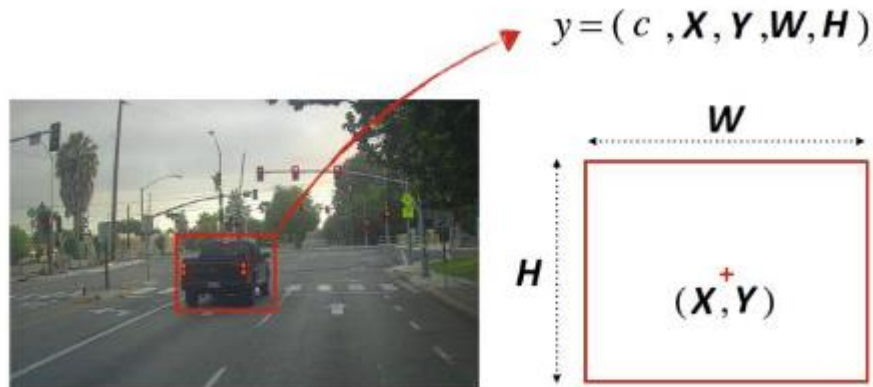
YOLO (*engl. You Only Look Once*) je model za detekciju objekta u realnom vremenu predstavljen od strane Joseph Redmon, Santosh Divvala, Ross Girshick, i Ali Farhadi, 2015. godine. Glavna ideja ovog modela je da zadatak detekcije i klasifikacije obavi na isti način na koji to ljudi rade, gledajući u sliku samo jednom. Ovaj metod zahteva samo jednu propagaciju unapred kroz neuronsku mrežu kako bi predvideo lokaciju objekta i dodelio kategoričku labelu, što ga čini izuzetno brzim. U realnom vremenu može da obradi 45 slika u sekundi, dok manja verzija ove mreže može da obradi 155 slika u sekundi [24].

YOLO radi tako što ulaznu sliku deli na mrežu od $S \times S$ kvadrata. Onaj kvadrat koji sadrži centar objekta se smatra odgovornim za predikciju tog objekta (Slika 18).



Slika 18 - Kvadrat koji ima najveću IoU vrednost sa *ground truth* se smatra odgovornim za predikciju objekta. Preuzeto iz [36].

Svaki kvadrat vrši predikciju B graničnih okvira i odgovarajućih vrednosti pouzdanosti (*engl. confidence score*). Predikcija jednog graničnog okvira (*engl. bounding box*) se sastoji od 5 komponenti (*confidence score*, x , y , w , h). Koordinate x i y predstavljaju centar graničnog okvira. w i h predstavljaju širinu i visinu graničnog okvira u odnosu na celu sliku (Slika 19). Ove koordinate su normalizovane tako da njihove vrednosti budu u opsegu od 0 do 1. *Confidence score* je definisan kao $\Pr(\text{Object}) * IOU_{pred}^{truth}$ i predstavlja verovatnoću da kvadrat sadrži objekat. Ukoliko u tom kvadratu nema objekta, *confidence score* je jednak nuli, u suprotnom je jednak preseku između prediktovanog graničnog okvira i *ground truth* lokacije (IoU).



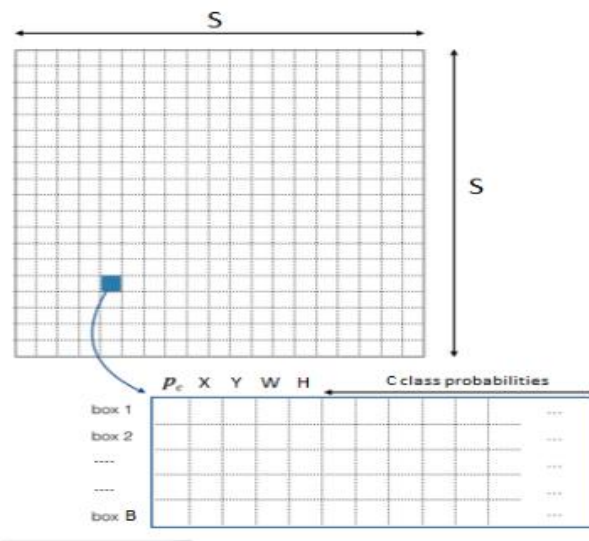
Slika 19 – (x,y) su koordinate centra, (w,h) su širina i visina graničnog okvira, c predstavlja verovatnoću (*confidence score*) da kvadrat sadrži objekat. Preuzeto iz [36].

Ukoliko kvadrat sadrži objekat, za svaku od mogućih klasa C se vrši predikcija verovatnoće $\Pr(Class_i | Object)$ da objekat pripada određenoj klasi. Ova verovatnoća se zove *conditional class probability* jer zavisi od postojanja objekta u kvadratu. *Confidence score* za svaku od C klasa se računa množenjem individualnih *confidence score* sa *conditional class probability*:

$$\Pr(Class_i | Object) * \Pr(Object) * IOU_{pred}^{truth} = \Pr(Class_i) * IOU_{pred}^{truth}$$

Dobijena vrednost nam govori verovatnoću da granični okvir sadrži objekat koji pripada određenoj klasi i koliko je dobro prediktovana lokacija graničnog okvira.

Jedna slika sadrži $S \times S \times B$ predikcija graničnih okvira, gde je svaka od tih predikcija sačinjena od 4 broja koji predstavljaju predikciju lokacije x,y,w,h i jednog broja koji predstavlja *confidence score* i C *conditional class probability* za klasifikaciju objekta (Slika 20).



Slika 20 - Ulazna slika se deli na $S \times S$ kvadrata. Svaki kvadrat daje B predikcija koje se sastoje od prediktovane lokacije (x,y,w,h) , *confidence score* (p_c) i C *conditional class probability*. Preuzeto iz [36].

Tokom treniranja se optimizuje sledeća funkcija greške koja se može podeliti na tri dela:

$$\begin{aligned}
& \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbf{1}_{ij}^{obj} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] \\
& + \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbf{1}_{ij}^{obj} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right] \\
& + \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbf{1}_{ij}^{obj} (C_i - \hat{C}_i)^2 + \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbf{1}_{ij}^{noobj} (C_i - \hat{C}_i)^2 \\
& + \sum_{i=0}^{S^2} \mathbf{1}_i^{obj} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2.
\end{aligned}$$

Localization loss:

$$\begin{aligned}
& \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbf{1}_{ij}^{obj} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] \\
& + \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbf{1}_{ij}^{obj} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right].
\end{aligned}$$

Ova jednačina računa grešku koja se odnosi na poziciju prediktovanog graničnog okvira. Funkcija računa sumu greški za svaki granični okvir (ukupno B okvira) za svaki kvadrat (ukupno $S \times S = S^2$). (\hat{x}, \hat{y}) predstavlja prediktovanu poziciju graničnog okvira a (x, y) je stvarna pozicija iz trening skupa podataka.

$\mathbf{1}_{ij}^{obj}$ ima vrednost 1 ukoliko je objekat prisutan u kvadratu i i j -ti granični okvir je odgovoran za njegovu predikciju, u suprotnom ima vrednost 0. Drugi deo ove jednačine se odnosi na prediktovanu visinu i širinu graničnog okvira. Kvadratni koren se koristi kako bi se granični okvir manjih dimenzija više penalizovao.

λ parametar koji se pojavljuje u funkciji greške se koristi da kontroliše koliko želimo da povećamo grešku od predikcija graničnog okvira (λ_{coord}) i koliko želimo da smanjimo grešku od *confidence score* predikcija za granične okvire koji ne sadrže objekte (λ_{noobj}). U originalnom radu autori koriste $\lambda_{coord} = 5$ i $\lambda_{noobj} = 0.5$ [24].

Confidence loss:

$$\sum_{i=0}^{S^2} \sum_{j=0}^B \mathbf{1}_{ij}^{obj} (C_i - \hat{C}_i)^2 + \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbf{1}_{ij}^{noobj} (C_i - \hat{C}_i)^2.$$

Ova jednačina računa grešku koja se odnosi na *confidence score* za svaku predikciju graničnog okvira. \hat{C}_i je *confidence score* za kvadrat i a C_i je količnik preseka i unije prediktovanog graničnog okvira sa njegovom stvarnom lokacijom (IoU). Jednačina pokušava da *confidence score* učini jednakim sa IoU između stvarne pozicije objekta i njegove predikcije samo kada u tom kvadratu postoji jedan objekat.

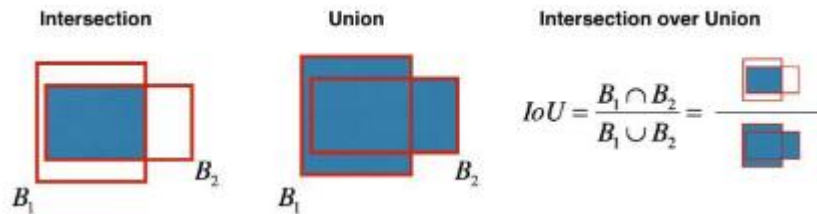
1_{ij}^{obj} ima vrednost 1 ukoliko je objekat prisutan u kvadratu i i j -ti granični okvir je odgovoran za njegovu predikciju, u suprotnom ima vrednost 0. Drugi deo jednačine pokušava da *confidence score* postavi na 0 kada u tom kvadratu ne postoji objekat.

Classification loss:

$$\sum_{i=0}^{S^2} 1_i^{obj} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2.$$

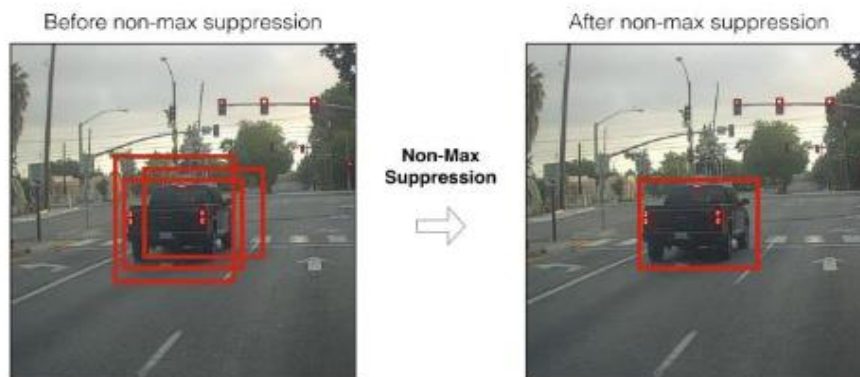
Ova jednačina računa kvadratnu sumu greške za klasifikaciju. 1_i^{obj} se koristi zato što funkcija greške ne bi trebalo da penalizuje klasifikacionu grešku kada objekat nije prisutan u kvadratu (ima vrednost 1 kada je objekat prisutan u i -tom kvadratu, vrednost 0 u suprotnom). $p_i(c)$ je *conditional class probability* koja nam govori da li kvadrat i sadrži objekat klase $c \in C$.

Kako YOLO daje više predikcija graničnih okvira za jedan kvadrat, koristi se metod IoU da oceni lokalizaciju objekta za svaki prediktovani granični okvir kako bi samo jedan granični okvir bio odgovoran za predikciju objekta. Računa se tako što se presek graničnog okvira i njegove stvarne pozicije podeli sa njihovom unijom. Granični okvir sa najvećom IoU vrednošću je odgovoran za predikciju tog objekta (Slika 21).



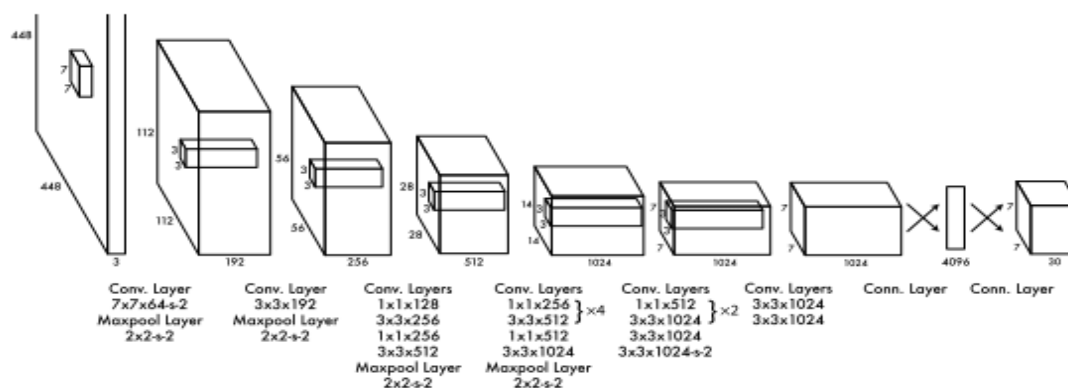
Slika 21 - *Intersection over Union (IoU)*. Preuzeto iz [36].

YOLO može da napravi predikcije koje se preklapaju za isti objekat (Slika 22). *Non-maximum suppression* je algoritam kojim se uklanjaju duplikati predikcija koje poseduju sa manjom pouzdanošću. *Non-maximum suppression* prvo sortira predviđanja po vrednostima *confidence score*. Nakon toga kreće od najboljih rezultata i zanemaruje bilo koju trenutnu predikciju ako pronađe bilo koju prethodnu predikciju koja ima istu klasu i $\text{IoU} > 0.5$ sa trenutnom predikcijom.



Slika 22 - *Non-maximum suppression* - dolazi do uklanjanja nepotrebnih predikcija. Preuzeto iz [36].

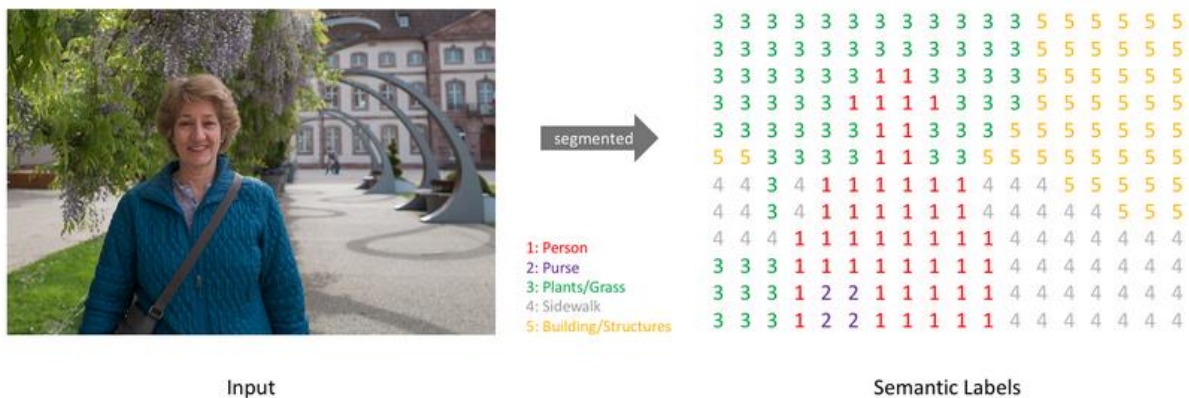
Arhitektura YOLO modela je inspirisana *GoogLeNet* modelom za klasifikaciju slika [27]. Model ima 24 konvoluciona sloja, koji su prethodno obučeni na *ImageNet* skupu podataka [6], nakon kojih su dva potpuno povezana sloja. Poslednji sloj predviđa verovatnoću klase i koordinate graničnih okvira (Slika 23).



Slika 23 - YOLO arhitektura modela. Preuzeto iz [24].

3.5 Segmentacija slika

Semantička segmentacija slike za zadatak ima da izvrši klasifikaciju svakog piksela slike, tj. da za ulaznu sliku dimenzija $W \times H \times 3$ generiše izlaznu matricu dimenzije $W \times H$ gde svaki element predstavlja identifikator klase kojoj odgovarajući piksel na ulaznoj slici pripada (Slika 24).



Slika 24 - Zadatak segmentacije slike je klasifikovati svaki piksel ulazne slike. Preuzeto iz [37].

Neke od primena semantičke segmentacije su: segmentacija medicinskih slika koja lekarima pomaže pri dijagnostici bolesti, segmentacija slika puteva što pronalazi primenu u autonomnim vozilima, segmentacija različitih tipova zemlje iz satelitskih slika (Slika 25).



Slika 25 - Segmentacija MRI slika, puteva i satelitskih slika. Preuzeto iz [37].

Neki od prvobitnih pristupa segmentacije slike su *region-based*, *edge-based* i segmentacija klasterovanjem dok su novije metode bazirane na konvolucionim neuronskim mrežama [28].

Region-based segmentacija

Kod *region-based* segmentacije se slika segmentira na osnovu vrednosti piksela, obzirom da znamo da će na ivicama objekata postojati velika razlika u vrednostima piksela. Ukoliko razlika vrednosti susednih piksela bude manja od postavljene vrednosti praga, onda tu regiju možemo segmentirati. U slučaju preklapanja objekata, potrebno je definisati više različitih vrednosti za prag. Mana ovog metoda je to što se definicija praga obavlja ručno.

Edge-based segmentacija

Osnovna ideja ove metode je da su ivice ono što deli različite objekte. Kada se prilikom obrade slike naiđe na ivicu, smatra se da je pronađen novi objekat koji se može segmentirati. Otkrivanje ivica se vrši upotrebom filtera i konvolucija koje prelaze preko slike i pronalaze ivice. Mana ove metode je ta što se filter za različite tipove ivica (npr. horizontalne ili vertikalne ivice) definiše ručno.

Segmentacija klasterovanjem

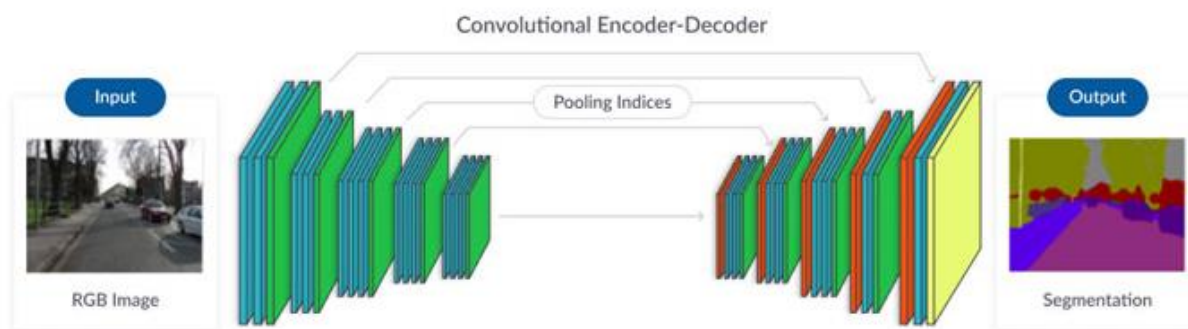
Kod segmentacije klasterovanjem se vrši grupisanje sličnih piksela u iste klustere. Potrebno je odabrati broj klastera koji predstavljaju broj objekata koji su prisutni na slici. Tokom klasterovanja se slični pikseli

smeštaju i isti klaster, nakon čega se vrši segmentacija u nekoliko regiona. Metoda daje dobre rezultate na malim i jednostavnim skupovima podataka [44].

Manu ovih metoda predstavlja potreba za ručnim podešavanjima parametara za specifične slučajeve i ograničena preciznost na kompleksnijim slikama. Prednost je jednostavnost implementacije, brzina i efikasnost segmentacije [28].

Segmentacija zasnovana na konvolucionim neuronskim mrežama

Konvolucione neuronske mreže su se pokazale dobrim metodama za segmentaciju slika jer imaju mogućnost da iz slike izdvoje veliki broj karakteristika i osobina [45]. Osnovna struktura *CNN based* modela za segmentaciju se sastoji od *encoder-a*, *decoder-a* i *skip* konekcija (Slika 26).



Slika 26 - Osnovna struktura *CNN-based* modela za segmentaciju slika. Preuzeto iz [38].

- *Encoder* - Predstavlja niz slojeva koji za zadatak imaju da smanje dimenziju ulaza i izdvoje bitne osobine ulazne slike. Za enkoder se često koristi unapred trenirana konvoluciona neuronska mreža koja je obučena za klasifikaciju slika, tako da već ima mogućnost prepoznavanja i izdvajanja bitnih karakteristika iz slike.
- *Decoder*: Predstavlja niz slojeva koji preuzimaju izlaz enkodera i povećavaju njegovu dimenziju i generišu segmentacionu masku klasifikovanih piksela.
- *Skip connections*: Predstavljaju vezu između izlaza enkodera sa ulazima dekodera na odgovarajućim pozicijama. Njihov zadatak je da nadoknade gubitak informacija između enkodera i dekodera.

Prilikom treniranja se minimizuje funkcija greške (obično *cross entropy loss*) koja svaki piksel izlaza mreže upoređuje sa odgovarajućim pikselom na slici koja predstavlja *ground truth* segmentacije.

U narednom poglavlju je opisana U-Net arhitektura koja je korišćena za segmentaciju ključnih tačaka ulazne slike (preseki linija fudbalskog terena). Ovaj model je odabran zbog svoje preciznosti, brzine treniranja i jer ne zahteva veliki skup podataka u procesu treniranja.

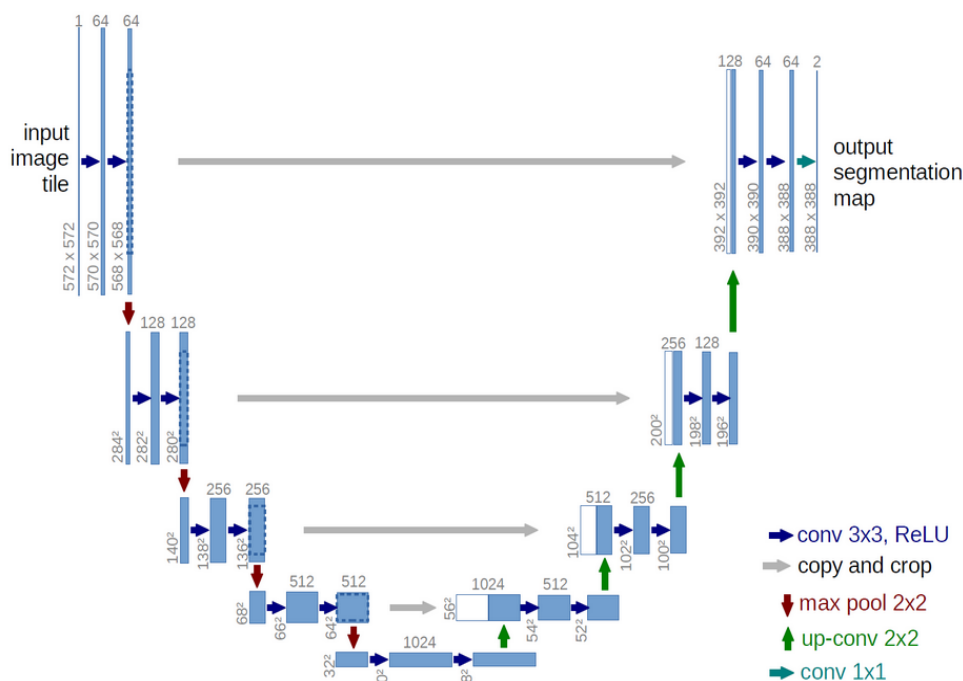
3.5.1 U-Net arhitektura

U-Net je model za segmentaciju slika prvobitno namenjen za segmentaciju biomedicinskih slika koji je 2015 osvojio ISBI izazov [29]. Arhitektura modela je bazirana na enkoder-dekoder strukturi [30]. Prednosti ovog modela su njegova brzina, jer može da izvrši klasifikaciju slika dimenzija 512×512 za samo jednu sekundu i mala količina podataka koja je potrebna za treniranje modela. Autori ovog modela

su sa samo 30 trening slika uspeali da treniraju model koji je osvojio pomenuti ISBI izazov. Arhitektura modela se može podeliti na dva dela: *encoder* (*contracting deo*) i *decoder* (*expansive deo*).

Encoder - Sastoji se od ponovljenih primena dve konvolucije dimenzije 3×3 koje su praćene *ReLU* aktivacionom funkcijom. Nakon toga se primenjuje 2×2 *max pooling* sloj kako bi se smanjile prostorne dimenzije. U svakom koraku se nakon smanjenja prostorne dimenzije za pola istovremeno duplira broj *feature* kanala kako bi model naučio kompleksnije osobine. *Encoder* je prikazan na levoj strani slike 27.

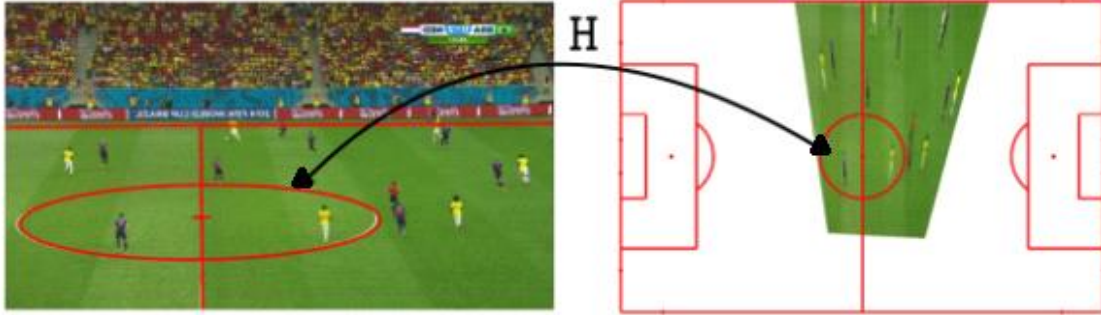
Decoder - U svakom koraku dekodera se primenjuje 2×2 konvolucija koja prepolovljava broj *feature* kanala dok istovremeno dolazi do povećanja prostorne dimenzije. Odgovarajuća *feature* mapa iz enkodera se spaja sa slojem u dekodera *skip* konekcijom kako bi se karakteristike naučene u enkoderu koristile za rekonstrukciju slike. Slede dva 3×3 konvoluciona sloja sa *ReLU* aktivacionom funkcijom. Na kraju se nalazi 1×1 konvolucija koja se koristi da mapira svaku komponentu *feature* vektora na željeni broj klasa. Ukupno, mreža ima 23 konvoluciona sloja (Slika 27). U primeru na slici je izlaz dimenzije 388×388 sa dva kanala od kojih svaki predstavlja jednu klasu koja postoji na slici.



Slika 27 - U-Net arhitektura se sastoji od enkodera (levog dela) i dekodera (desnog dela). Preuzeto iz [30].

3.6 Homografija

Homografija predstavlja transformaciju između dve ravni, tj. dve planarne projekcije slike (Slika 28).



Slika 28 – Transformacija između ulazne i izlazne slike upotrebom homografije.

Formalna definicija [31]:

Mapirajuća funkcija $h: P^2 \rightarrow P^2$ je homografija ako i samo ako postoji nesusingularna 3×3 matrica \mathbf{H} tako da za svaku tačku u u P^2 predstavljenu vektorom x važi da je $h(x) = \mathbf{H}x$

Primer: Neka su $(x_1, y_1), (x_2, y_2) \in \mathbb{R}^2$ i neka je $X_i = [x_i, y_i, 1]^T$. Planarna homografija predstavlja transformaciju između dve ravni definisane tačkama X_1 i X_2 i definiše se kao:

$$X_2 = \mathbf{H}X_1 = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} X_1, \quad \text{gde je } X_1 = \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}, X_2 = \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix}.$$

Kako bismo normalizovali matricu \mathbf{H} definišemo $h_{33}=1$. Iz jednačine slede deficije za x_2 i y_2 :

$$x_2 = \frac{h_{11}x_1 + h_{12}y_1 + h_{13}}{h_{31}x_1 + h_{32}y_1 + 1}$$

$$y_2 = \frac{h_{21}x_1 + h_{22}y_1 + h_{23}}{h_{31}x_1 + h_{32}y_1 + 1}.$$

Što se može pojednostaviti:

$$x_2 = h_{11}x_1 + h_{12}y_1 + h_{13} - h_{31}x_1x_2 - h_{32}y_1x_2$$

$$y_2 = h_{21}x_1 + h_{22}y_1 + h_{23} - h_{31}x_1y_2 - h_{32}y_1y_2$$

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1x_2 & -y_1x_2 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1y_2 & -y_1y_2 \end{bmatrix} h = \begin{bmatrix} x_2 \\ y_2 \end{bmatrix}$$

Gde je $h = [h_{11}, h_{12}, h_{13}, h_{21}, h_{22}, h_{23}, h_{31}, h_{32}]^T$.

Za par od n tačaka imamo sistem:

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1x_2 & -y_1x_2 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1y_2 & -y_1y_2 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ x_{n1} & y_{n1} & 1 & 0 & 0 & 0 & -x_{n1}x_{n2} & -y_{n1}x_{n2} \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_{n1}y_{n2} & -y_{n1}y_{n2} \end{bmatrix}_{2Nx8} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \end{bmatrix}_{8x1} = \begin{bmatrix} x_2 \\ y_2 \\ \dots \\ x_{n2} \\ y_{n2} \end{bmatrix}_{2Nx1}$$

$$A_{2Nx8} h_{8x1} = b_{2Nx1}$$

$$A_{8x2N}^T A_{2Nx8} h_{8x1} = A_{8x2N}^T b_{2Nx1}$$

$$(A^T A)_{8x8} h_{8x1} = (A^T b)_{8x1}$$

$$h_{8x1} = (A^T A)^{-1}_{8x8} (A^T b)_{8x1}$$

Za računanje homografije između dve ravni potrebno nam je najmanje 4 tačke za svaku ravan.

4. Specifikacija i implementacija rešenja

U ovom poglavlju će biti predstavljeni skup podataka korišćen za treniranje i evaluaciju modela (4.1) i specifikacija rešenja (4.2).

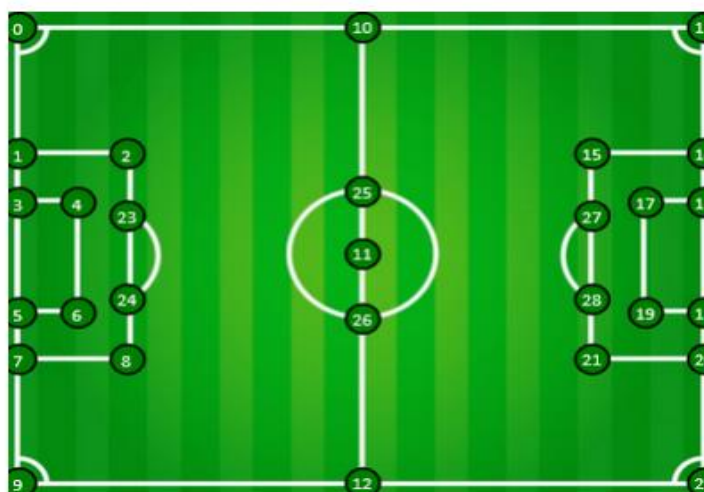
4.1 Skup podataka

Skup podataka korišćen u ovom radu predstavlja proširenje skupa iz [1] i sastoji se od 565 slika (dimenzija 320 x 320 x 3) sa svetskog prvenstva 2014. godine. Svaka slika ima svoj odgovarajući fajl koji sadrži podatke o koordinatama svake od 29 ključnih tačaka koje su vidljive na slici (Slika 29), kao i koordinate graničnih okvira igrača na slic. Podaci su podeljeni na tri podskupa (trening, validacioni i test) u razmeri 80:10:10.



Slika 29 - Primer slika sa obeleženim ključnim tačkama koje su opisane x i y koordinatama.

Svakoj ključnoj tački koja predstavlja presek linija je dodeljen jedinstveni broj koji prilikom segmentacije slike predstavlja klasu piksela (Slika 30).



Slika 30 - Prikaz 29 tačaka koje se izdvajaju kao ključne tačke prilikom segmentacije.

Svaki igrač je predstavljen koordinatama svog graničnog okvira, tj. koordinatama levog gornjeg i desnog donjeg ugla (Slika 31).

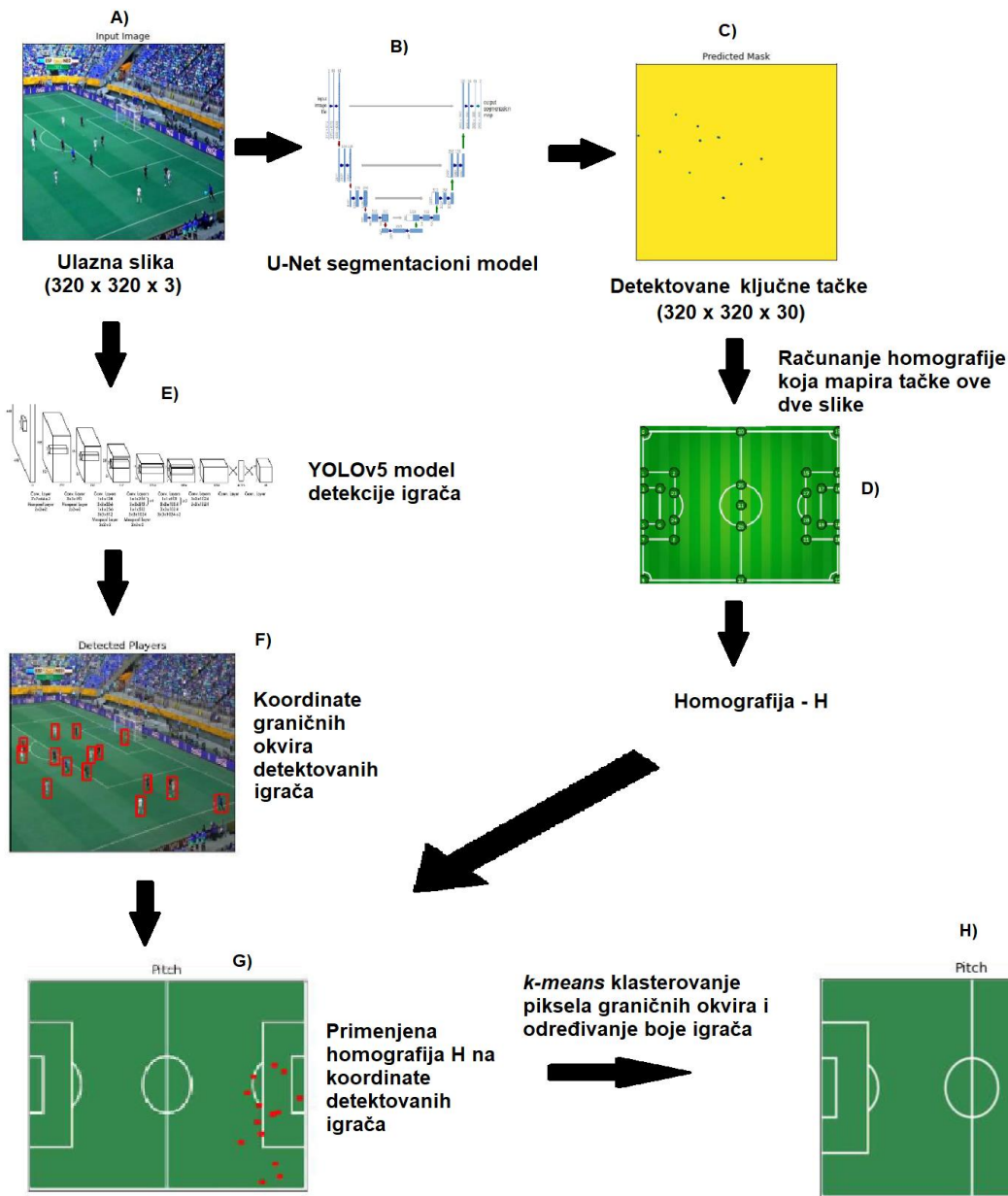


Slika 31 - Svaki igrač je opisan pozicijom levog gornjeg i desnog donjeg ugla njegovog graničnog okvira.

4.2 Specifikacija rešenja

Na slici 32 je predstavljen celokupno rešenje i njegovi delovi. Ulaz predstavlja slika dimenzije 320 x 320 x 3 (Slika 32.A) koja se dalje prosleđuje U-Net segmentacionom modelu (Slika 32.B). Zadatak U-Net modela je da detektuje ključne tačke i njih predstavi matricom dimenzije 320 x 320 x 30 što predstavlja izlaz modela (Slika 32.C). Mapiranjem koordinata detektovanih ključnih tačaka sa slike 32.C na njihove ciljne koordinate sa slike 32.D se računa homografija H. Ulazna slika se takođe prosleđuje YOLO modelu za detekciju igrača (Slika 32.E). Izlaz predstavljaju koordinate graničnih okvira detektovanih igrača (Slika 32.F). Primenom prethodno izračunate homografije H na koordinate graničnih okvira se računaju ciljne koordinate igrača (Slika 32.G). Kako bi se vizualizovali timovi kojima igrači pripadaju, primenjuje se *k-means* algoritam za klasterovanje piksela detektovanih graničnih okvira igrača i konačan izlaz je predstavljen na slici 32.H.

Modul za segmentaciju ulazne slike i određivanje homografije je predstavljen u poglavlju 4.2.1. U poglavlju 4.2.2 je opisan modul za detekciju igrača i transformaciju njihovih pozicija prethodno izračunatom homografijom, dok je modul za klasterizaciju piksela detektovanih igrača predstavljen u poglavlju 4.2.3.



Slika 32 – Ulazna slika (A) se prosleđuje U-Net modelu (B) koji za izlaz daje matricu koja predstavlja detektovane tačke koje su vizualizovane na slici (C). Mapiranjem tačaka na njihove ciljne koordinate sa slike (D), dobijamo homografiju H. Na sliku (A) se primenjuje YOLO (E) model za detekciju igrača (F). Primenom homografije na igrače dobijamo sliku (G). *k-means* klasterovanjem piksela dobijamo boje igrača za konačni prikaz (H).

Svi modeli su trenirani korišćenjem besplatne verzije *Google Colab*¹ alata čije su specifikacije:

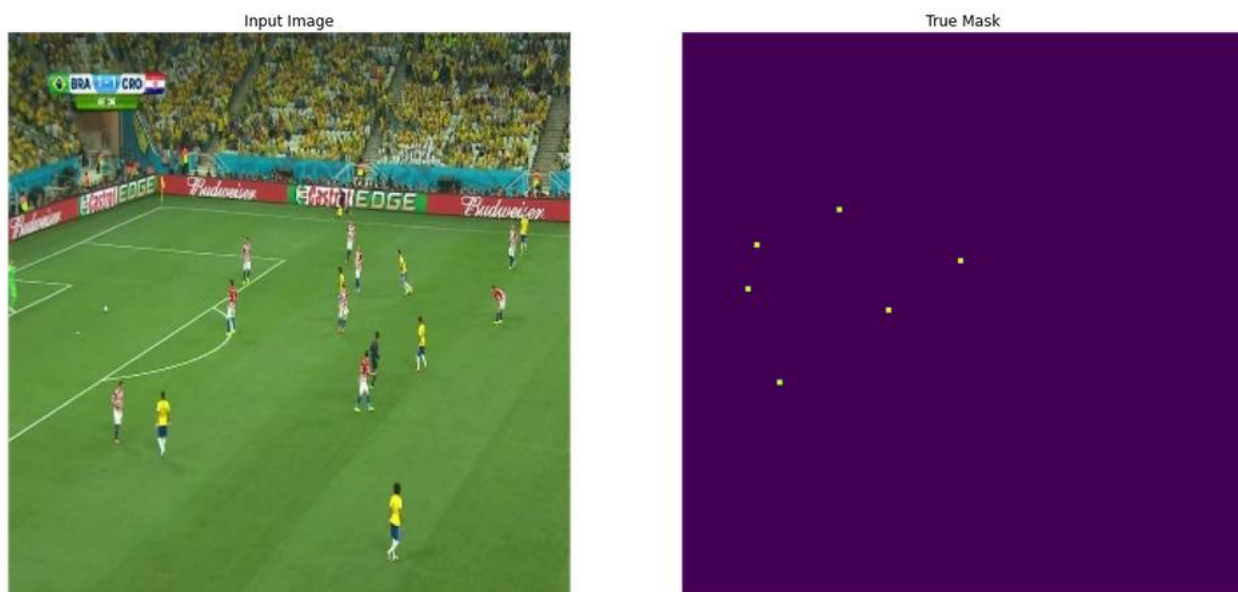
- CPU - Intel(R) Xeon(R) @ 2.20GHz.
- GPU - Tesla T4 13GB RAM.

¹ <https://colab.research.google.com/>

4.2.1 Segmentacija i detekcija ključnih tačaka

Zadatak modula za segmentaciju je da na ulaznoj slici detektuje ključne tačke i odredi kojoj klasi pripadaju od mogućih 29 tačaka.

Pre samog početka treniranja je potrebno transformisati podatke koje predstavljaju ključne tačke. Kako su oni zadati samo svojom koordinatom i klasom, potrebno je kreirati matricu širine i visine od 320 što odgovara ulaznoj slici i dubine 30 gde svaki sloj odgovara jednoj od klasa (ukupno 29 klasa za ključne tačke i jedna za pozadinu). Vrednost u matrici k -te dubine je 1 ukoliko se na toj poziciji u ulaznoj slici nalazi ključna tačka k -te klase. Na slici 33 je prikazan par ulazne slike i njene maske sa 6 vidljivih ključnih tačaka koje pripadaju klasama 24, 23, 22, 4, 1, 0.



Slika 33 - Ulazna slika (levo) i prikaz ključnih tačaka koje se na slici nalaze (desno).

Prilikom treniranja model na ulazu prima sliku dimenzije $320 \times 320 \times 3$ i matricu koja predstavlja masku dimenzije $320 \times 320 \times 30$.

Segmentacija je trenirana korišćenjem U-Net modela gde je za *backbone* enkodera korišćen *efficientNet* [39], *resnet34* [40] i *mobileNet* [41] koji su prethodno trenirani na *ImageNet* skupu podataka.

Za funkciju greške su testirane dve različite funkcije: *focal loss* i *dice loss*, čije su prednosti u tome što dobro procenjuju grešku za skup podataka koji ima neujednačen broj predstavnika svake klase [42]. U skupu podataka postoji dosta slika koje ne sadrže ključne tačke sredine terena (klase 10, 25, 11, 26, 12).

Dice loss meri presek između predikcije i stvarne vrednosti. \hat{p} predstavlja predikciju klase za svaki piksel, dok y predstavlja matricu gde je svakom pikselu dodeljena tačna klasa.

$$DL(y, \hat{p}) = 1 - \frac{2y\hat{p} + 1}{y + \hat{p} + 1}$$

$y\hat{p}$ predstavlja presek stvarnih i prediktovanih vrednosti:

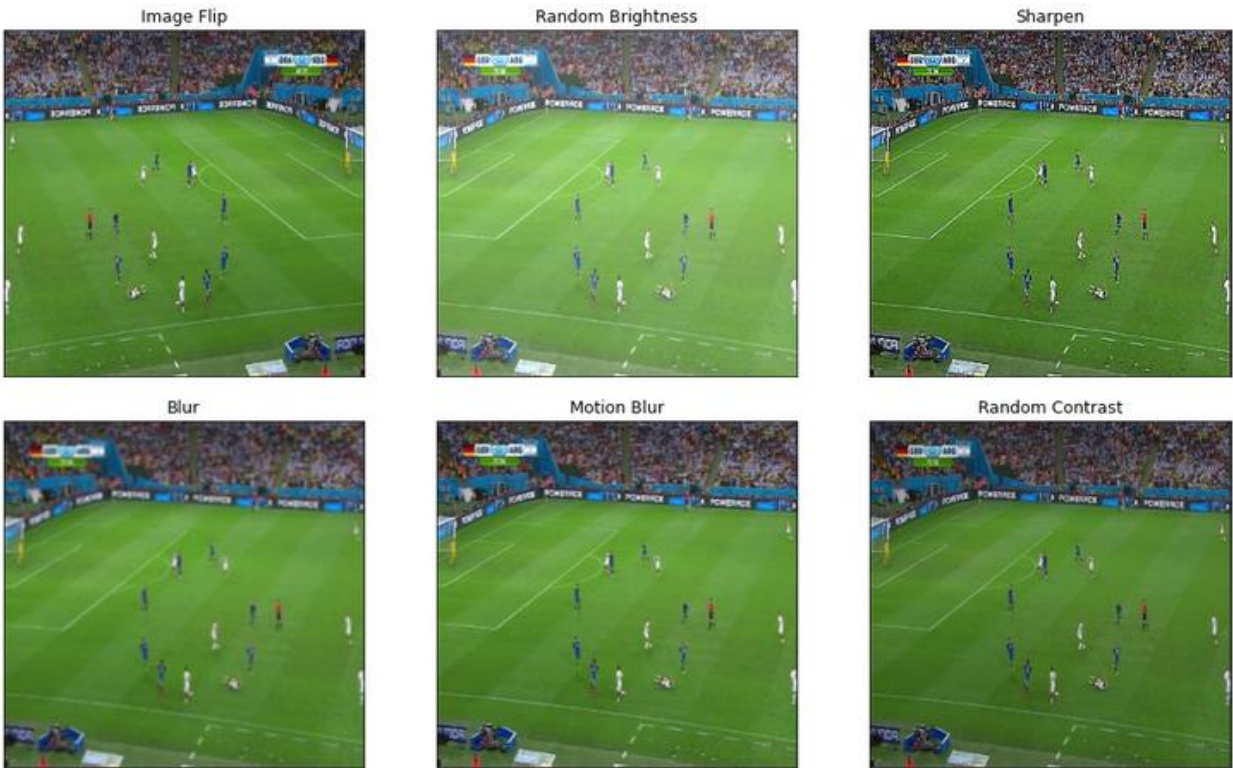
$$|y \cap \hat{p}| = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}_{target} * \begin{bmatrix} 0.01 & 0.03 & 0.02 \\ 0.05 & 0.12 & 0.09 \\ 0.89 & 0.85 & 0.88 \\ 0.99 & 0.97 & 0.95 \end{bmatrix}_{prediction} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0.89 & 0.85 & 0.88 \\ 0.99 & 0.97 & 0.95 \end{bmatrix} \xrightarrow{sum} 5.53.$$

Focal loss je varijacija *cross entropy loss*-a koji umanjuje doprinos lakih primera i omogućava modelu da se fokusira na učenje težih primera.

$$FL(p_t) = -\alpha(1 - p_t)^\gamma \log(p_t).$$

α i γ su hiperparametri. Veliko vrednost za γ smanjuje grešku za primere koje je lako klasifikovati, dok se fokus stavlja na teže primere i obično se uzima vrednost 2. α ima vrednost 0.25 i služi za balansiranje funkcije greške.

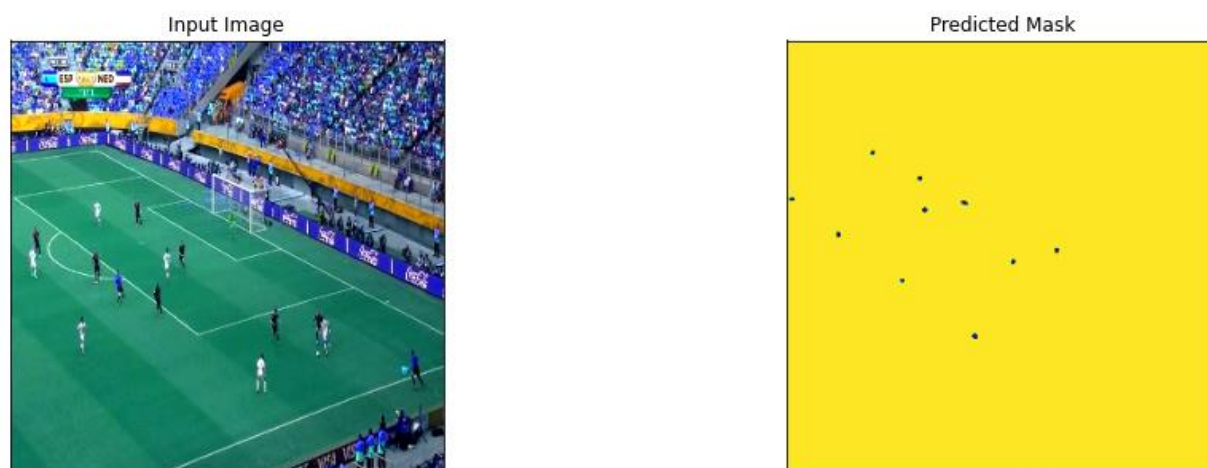
Prilikom treniranja se na ulazne slike primenjuje augmentacija slike: *image flip*, *random brightness*, *sharpen*, *blur*, *motion blur*, *random contrast* (Slika 34).



Slika 34 - Primeri augmentacija koje su primenjene na ulaznu sliku prilikom treniranja.

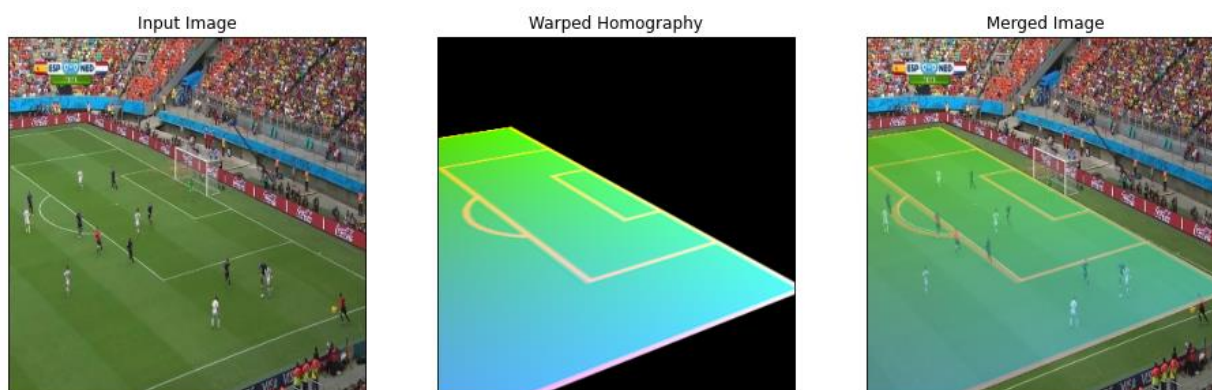
Prilikom treniranja segmentacionog modela su korišćeni parametri *batch size* 4, *learning rate* 0.0001, *Adam optimizator* i broj epoha treniranja je 100.

Izlaz modela za segmentaciju je matrica dimenzije 320 x 320 x 30. Prolaskom kroz svaku od 30 matrica dubine i filtriranjem vrednosti većih od 0.9 dobijamo lokaciju prediktovane ključne tačke i klasu kojoj pripada (Slika 35)².



Slika 35 - Prikaz ulazne slike i prediktovanih ključnih tačaka.

Znajući koordinate bar 4 ključne tačke, kojoj klasi pripadaju kao i njihove željene koordinate, može se izvršiti računanje homografije i transformacija ulazne slike (Slika 36).



Slika 36 - Na osnovu pronađenih tačaka se računa homografija. Primenom homografije na *template* sliku i kombinovanjem sa ulaznom se dobija slika desno.

4.2.2 Detekcija igrača

Zadatak ovog modula je da detektuje igrače kako bi se na njihove koordinate primenila homografija izračunata u prethodnom koraku i izvršila transformacija u drugi prostor.

Za detekciju je korišćen unapred treniran YOLO model [43] sa 21.4 miliona parametara koji je dodatno treniran na 449 slika skupa podataka objašnjenog u poglavlju 4.1. Za potrebe treniranja je bilo neophodno transformisati koordinate graničnog okvira tako da su u formatu (*class x_center y_center width height*), gde je:

² Ukoliko se u petoj matrici dubine na poziciji 150 x 250 nalazi vrednost 0.97 to znači da je piksel sa koordinatama (150, 250) nalazi ključna tačka kojoj je dodeljena klasa 5 sa slike 30.

- *class* identifikator klase koju objekat predstavlja (0 za čoveka),
- *x_center* i *y_center* centar graničnog okvira koji se normalizuje deljenjem koordinata sa širinom i visinom ulazne slike,
- *width* i *height* normalizovanu širinu i visinu graničnog okvira.

Model je treniran 50 epoha sa parametrom *batch size* 16. Kao konačna koordinata koju je primenom homografije potrebno transformisati se uzima sredina osnove graničnog okvira.

Na slici 37 je prikazan primer detekcije igrača sa ulazne slike i primena homografije za transformaciju u 2D prostor.

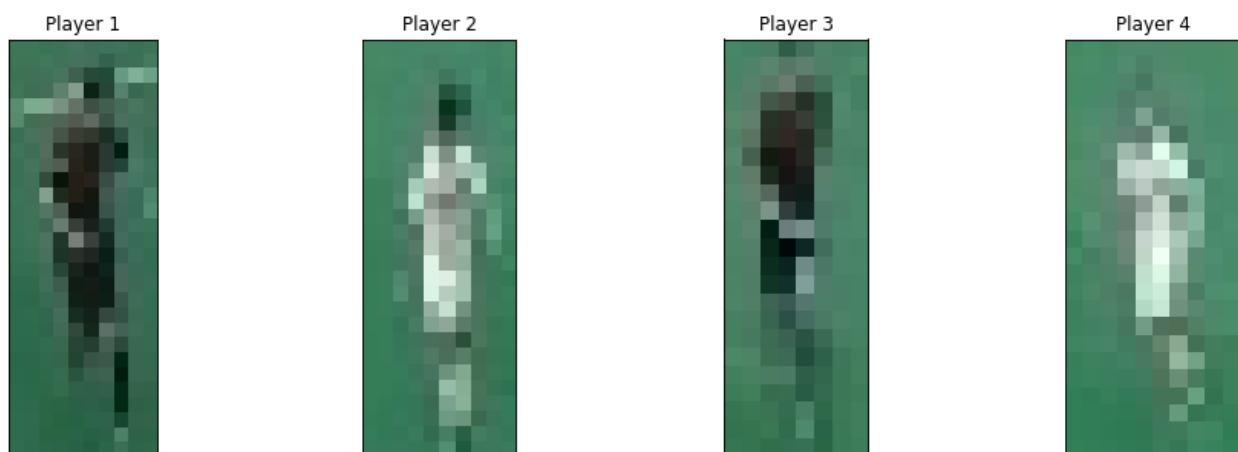


Slika 37 - Primenom YOLO modela se vrši detekcija igrača i primena homografije

4.2.3 Klasterizacija igrača

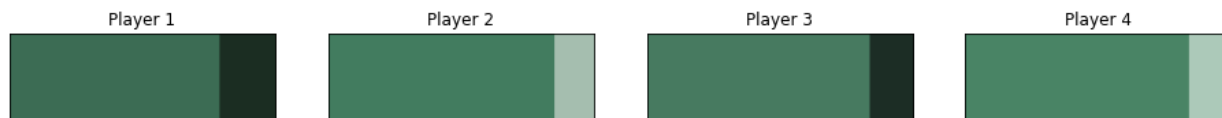
Zadatak modula za klasterizaciju igrača je da preuzme vrednosti piksela svakog detektovanog igrača i da odredi boju dresa koju igrač nosi što se koristi za vizuelni prikaz pozicija igrača.

Ulaz u modul su koordinate levog gornjeg ugla graničnog okvira ($x1, y1$) i desnog donjeg ugla ($x2, y2$) za svaku detekciju na ulaznoj slici. Nakon toga se vrši izdvajanje regija slike koje sadrže igrače (Slika 38).



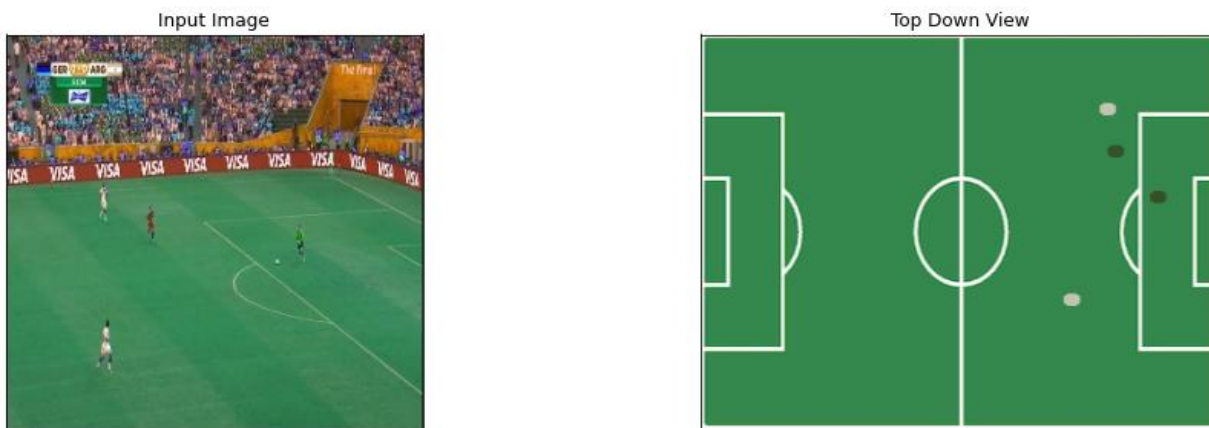
Slika 38 - Izdvojene regije slike na kojima su detektovani igrači.

Primenom *k-means* algoritma za klasterizaciju sa dva klastera na izdvojene regije se mogu izdvojiti dve najdominantnije boje na svakoj regiji (Slika 39). Vrednost boje predstavlja centar klastera.



Slika 39 - Prikaz boja nakon klasterovanja izdvojenih regija sa igračima.

Obzirom da se zna da će u velikom broju slučajeva najdominantnija boja u svakoj detektovanoj regiji igrača biti zelena boja koja predstavlja pozadinu, za boju koja predstavlja igrača se uzima druga najzastupljenija boja. Nakon što se svakoj regiji dodeli boja koja predstavlja igrača, potrebno je sve regije igrača ponovo klasterizovati u dva klastera obzirom da svaki igrač jednog tima treba biti obojen istom bojom. Konačan izlaz ovog modula se dobija primenom homografije na koordinate detektovanih igrača i prikaz na slici odgovarajućim bojama (Slika 40).



Slika 40 - Prikaz detektovanih igrača označenih bojama svog tima.

5. Evaluacija rešenja i rezultati

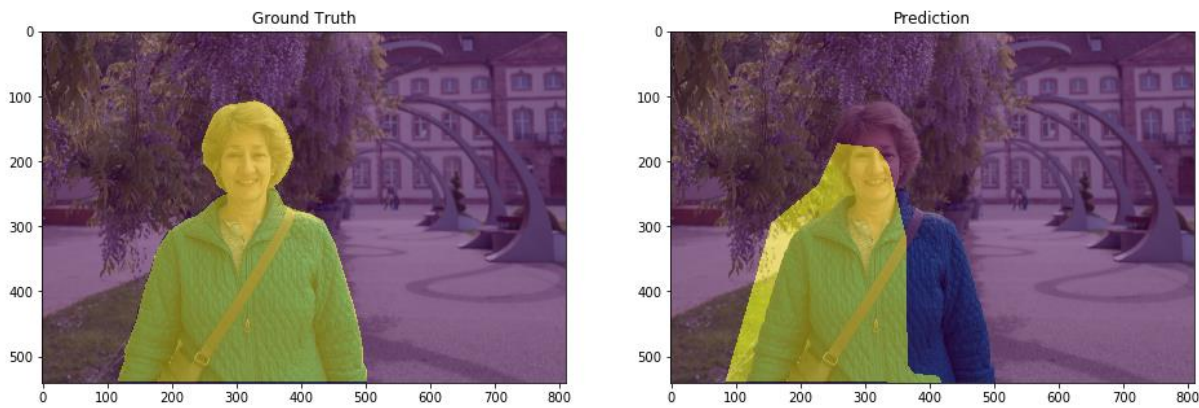
Skup podataka opisan u poglavlju 4.2 je podeljen na tri dela: trening, validacioni i test skup u razmeri 80:10:10, tj. 449 slika za trening i po 58 slika za validacioni i testni skup podataka. Odabir konačnog modela, hiperparametara i funkcija greške je rađen na osnovu performansi modela na validacionom skupu dok je konačna evaluacija odrađena na testnom skupu podataka. U poglavlju 5.1 je opisan način evaluacije i rezultati modela segmentacije, dok je evaluacija modela detekcije igrača sa ostvarenim rezultatima predstavljena u poglavlju 5.2.

5.1 Evaluacija modela segmentacije

Za evaluaciju modela segmentacije je, kao i u rešenjima [2], [3] i [5] opisanim u poglavlju 2., korišćena metrika IoU (engl. *Intersection over Union*) koja se definiše kao količnik piksela zajedničkih za prediktovanu i ciljnu masku i njihove unije:

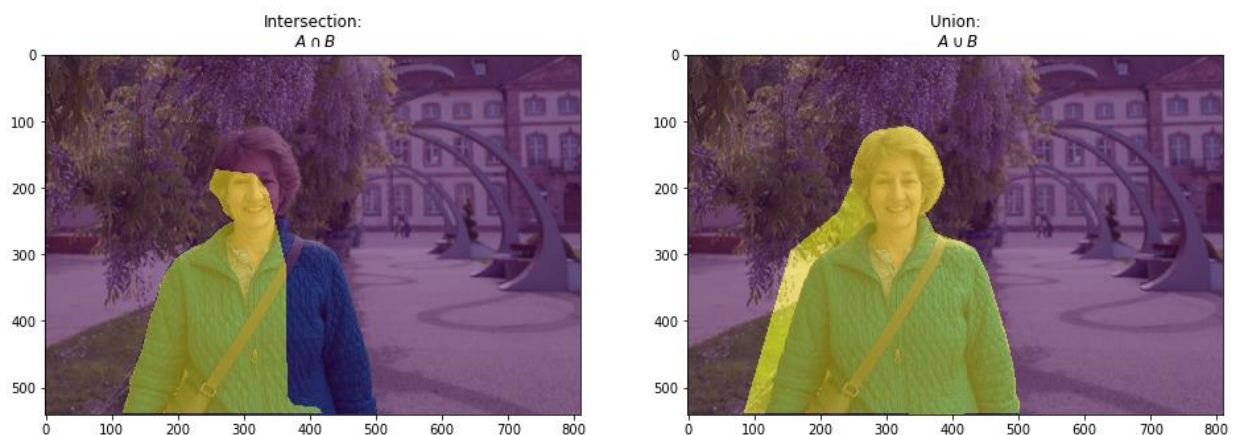
$$IoU = \frac{target \cap prediction}{target \cup prediction}$$

Na slici 41 je prikazana ciljna maska (levo) i prediktovana maska (desno).



Slika 41 - Primer ciljne (levo) i prediktovane (desno) maske. Preuzeto u [37].

Presek se definiše kao skup piksela koji se nalaze i u ciljnoj i prediktovanoj maski dok je unija definisana kao skup svih piksela koji se nalaze u obe maske (Slika 42).



Slika 42 - Primer preseka i unije ciljne i prediktovane maske. Preuzeto [37].

IoU vrednost se kreće između 0 i 1. Vrednosti bliže 1 predstavljaju dobru predikciju maske a za prihvatljivu predikciju se uzimaju one koje imaju IoU vrednost veću od 0.5.

Backbone	Validation IoU	Validation Loss	Test IoU	Test Loss	Total parameters
efficientnet	0.8234	0.1637	0.8076	0.18305	17.872.038
mobilenet	0.76989	0.24824	0.77452	0.242	8.051.646
resnet34	0.83314	0.14984	0.82991	0.15342	24.460.359

Tabela 1 - Rezultati modela segmentacije na validacionom i test skupu.

U tabeli 1 su prikazani rezultati na validacionom i test skupu podataka sa različitim modelima enkodera. Najbolje rezultate na test skupu podataka je ostvario U-Net model sa *resnet34* enkoderom treniranom na *ImageNet* skupu podataka (IoU=**0.82991**). Model je treniran 100 epoha, learning rate 0.0001, adam optimizatorom i funkcijom greške koja predstavlja zbir *focal* i *dice loss*. Model ima ukupno 24.460.359 parametara i treniranje modela traje do 120 sekundi po epohi. Model je treniran sa *batch size* vrednošću 4 zbog ograničenja memorije.

Na sličnom skupu podataka su [3] i [5] ostvarili rezultat od 0.83 IoU dok je [2] ostvario rezultat od 0.87. Test skup na kojem su pomenuti radovi izvršili testiranje se sastoji od 209 trening i 186 test slika i nema validacionog skupa. Skup korišćen u ovom radu predstavlja njegovo proširenje i sastoji se od 449 trening slika, 58 validacionih i test slika.

5.2 Evaluacija modela detekcije igrača

Za evaluaciju modela detekcije igrača je korišćena metoda *mAP* (engl. *mean average precision*) koja se najčešće koristi za evaluaciju i poređenje modela detekcije i klasifikacije objekata.

Preciznost (engl. *precision*) i odziv (engl. *recall*) su metrike koje se koriste za evaluaciju klasifikacionih modela i sastavni su deo *mAP* metrike.

Preciznost date klase u klasifikaciji je definisana kao odnos pravih pozitivnih (*True Positive*, *TP*) i ukupnog broja pozitivnih predikcija (*True Positive (TP) + False Positive (FP)*).

$$precision = \frac{TP}{TP + FP} = \frac{TP}{No. of predictions}$$

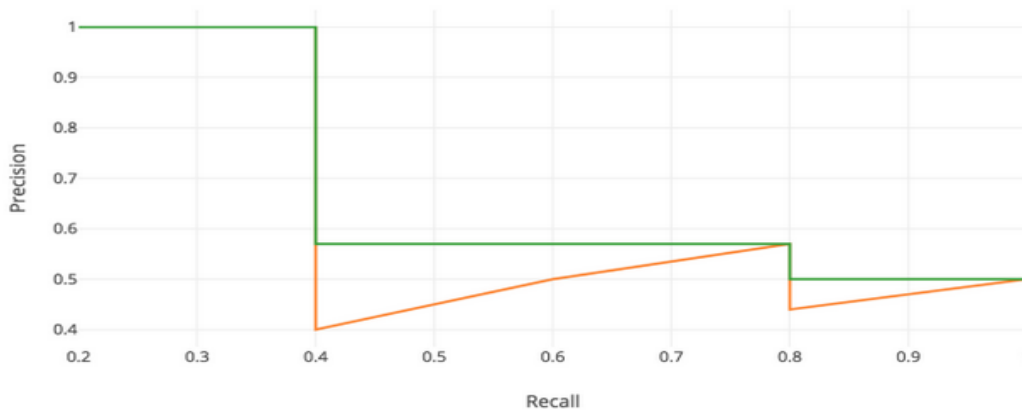
Odziv se definiše kao odnos pravih pozitivnih (*True Positive*, *TP*) i ukupnog broja pozitivnih primera (*True Positive* (*TP*) + *False Negative* (*FN*)):

$$recall = \frac{TP}{TP + FN} = \frac{TP}{No. of ground truths}$$

Predikcija se definiše kao *True Positive* ukoliko ima *IoU* vrednost veću od postavljene granice (npr. 0.5).

Slika 43 predstavlja *precision recall* krivu koja se dobija tako što se sve predikcije objekata na slikama sortiraju prema njihovoj pouzdanosti. Zatim, idući od najpouzdanije predikcije se računaju vrednosti za preciznost i odziv koje su prikazane na grafiku narandžastom bojom. Nakon toga se svaka vrednost za preciznost zameni sa maksimalnom vrednošću za preciznost desno od posmatrane vrednosti (zelena linija na grafiku). *Average Precision* (*AP*) se računa kao prosečna vrednost površine ispod zelene krive podeljene na 11 podeoka (0, 0.1, ..., 1).

Mean Average Precission (*mAP*) se računa kao prosečna vrednost *Average Precision* vrednosti primenjene na svaku od postojećih klasa.



Slika 43 - *Precision Recall* kriva

YOLO model je prvobitno treniran na COCO [32] skupu podataka i prepoznaje 80 različitih klasa. Dodatno je treniran na skupu podataka opisanom u poglavlju 4.2 za prepoznavanje klase *person* ukupno 50 epoha. Model je na testnom skupu podataka ostvario *mAP* od 0.9886 za prepoznavanje klase *person* dok je prosečna *IoU* vrednost predikcije 0.84.

Poređenje modela detekcije igrača sa radovima iz poglavlja 2 nije obrađeno obzirom da autori rada [3] nisu vršili evaluaciju modela detekcije igrača.

6. Zaključak

U ovom radu je predstavljena implementacija rešenja koja može poslužiti za prikupljanje *Tracking* podataka koristeći samo slike prenosa fudbalske utakmice što bi značajno smanjilo troškove prikupljanja podataka koji imaju veliku primenu u fudbalskoj analitici. Korišćen je skup podataka koji se sastoji od slika televizijskog prenosa svetskog prvenstva iz 2014 godine [1]. Sistem je implementiran u vidu tri odvojena modula: modula za segmentaciju ključnih tačaka ulazne slike, modula za detekciju igrača ulazne slike i modula klasterovanja piksela detektovanih regija.

Za segmentacioni model je korišćena U-Net arhitektura, koja je trenirana na testnom skupu podataka da detektuje 29 ključnih tačaka (preseki linija fudbalskog terena) [30]. Na osnovu detektovanih tačaka se računa homografija koja mapira ulaznu sliku na dvodimenzionalni prostor koji predstavlja *top down* pregled fudbalskog terena. Model je evaluiran na testnom skupu i ostvario rezultat od 83% IoU.

Detekcija igrača je implementirana korišćenjem YOLO modela za detekciju objekata koji je dodatno treniran na skupu podataka koji je korišćen u ovom radu [26]. Konačne pozicije igrača u izlaznoj slici se računaju transformisanjem koordinata detektovanih igrača homografijom. Model je postigao rezultat od 0.98 mAP.

Konačan prikaz boja igrača je odrađen korišćenjem *k-means* algoritma kako bi se klasterizovali pikseli detektovanih graničnih okvira igrača.

Svi modeli su trenirani uz korišćenje *Google Colab* alata što je predstavljalo najveće ograničenje preciznosti modela. Prilikom treniranja modela segmentacije je model bio ograničen na *batch size* 4 zbog nemogućnosti treniranja sa više memorije. Dimenzije ulaznih slika od 320 x 320 predstavljaju još jedno ograničenje obzirom da su lošijeg kvaliteta što otežava detektovanje ključnih tačaka. U realnoj primeni bi model za ulaz imao slike većih dimenzija što nije bilo moguće testirati obzirom na ograničenje memorije korišćenog alata. Dimenzije slika takođe utiču i na klasterizaciju piksela jer je zbog male dimenzije slika izgubljeno dosta piksela koji bi doprineli boljoj klasterizaciji prisutnih boja što se i vidi na slici 38.

7. Literatura

- [1] Homayounfar, N., Fidler, S. and Urtasun, R., 2017. Sports field localization via deep structured models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 5212-5220).
- [2] Chen, J. and Little, J.J., 2019. Sports camera calibration via synthetic data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops* (pp. 0-0).
- [3] Sharma, R.A., Bhat, B., Gandhi, V. and Jawahar, C.V., 2018, March. Automated top view registration of broadcast football videos. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)* (pp. 305-313). IEEE.
- [4] Isola, P., Zhu, J.Y., Zhou, T. and Efros, A.A., 2017. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1125-1134).
- [5] Jiang, W., Higuera, J.C.G., Angles, B., Sun, W., Javan, M. and Yi, K.M., 2020. Optimizing through learned errors for accurate sports field registration. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision* (pp. 201-210).
- [6] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M. and Berg, A.C., 2015. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3), pp.211-252.
- [7] Fukushima, K., 1988. Neocognitron: A hierarchical neural network capable of visual pattern recognition. *Neural networks*, 1(2), pp.119-130.
- [8] Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," in *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278-2324, Nov. 1998, doi: 10.1109/5.726791.
- [9] Janai, J., Güney, F., Behl, A. and Geiger, A., 2020. Computer vision for autonomous vehicles: Problems, datasets and state of the art. *Foundations and Trends® in Computer Graphics and Vision*, 12(1-3), pp.1-308.
- [10] Wang, M. and Deng, W., 2018. Deep face recognition: A survey. *arXiv preprint arXiv:1804.06655*.
- [11] M. Asadi-Aghbolaghi *et al.*, "A Survey on Deep Learning Based Approaches for Action and Gesture Recognition in Image Sequences," *2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017)*, 2017, pp. 476-483, doi: 10.1109/FG.2017.150.
- [12] Zhao, Z.Q., Zheng, P., Xu, S.T. and Wu, X., 2019. Object detection with deep learning: A review. *IEEE transactions on neural networks and learning systems*, 30(11), pp.3212-3232.
- [13] Ghofrani, J., Kirschne, R., Rossburg, D., Reichelt, D. and Dimter, T., 2019. Machine Vision in the Context of Robotics: A Systematic Literature Review. *arXiv preprint arXiv:1905.03708*.
- [14] O'Mahony, N., Campbell, S., Carvalho, A., Harapanahalli, S., Hernandez, G.V., Krpalkova, L., Riordan, D. and Walsh, J., 2019, April. Deep learning vs. traditional computer vision. In *Science and Information Conference* (pp. 128-144). Springer, Cham.
- [15] Girshick, R., Donahue, J., Darrell, T. and Malik, J., 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 580-587).
- [16] He, K., Zhang, X., Ren, S. and Sun, J., 2015. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 37(9), pp.1904-1916.

- [17] Girshick, R., 2015. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision* (pp. 1440-1448).
- [18] Ren, S., He, K., Girshick, R. and Sun, J., 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, pp.91-99.
- [19] Dai, J., Li, Y., He, K. and Sun, J., 2016. R-fcn: Object detection via region-based fully convolutional networks. In *Advances in neural information processing systems* (pp. 379-387).
- [20] He, K., Gkioxari, G., Dollár, P. and Girshick, R., 2017. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision* (pp. 2961-2969).
- [21] Erhan, D., Szegedy, C., Toshev, A. and Anguelov, D., 2014. Scalable object detection using deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2147-2154).
- [22] Yoo, D., Park, S., Lee, J.Y., Paek, A.S. and So Kweon, I., 2015. Attentionnet: Aggregating weak directions for accurate object detection. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 2659-2667).
- [23] Najibi, M., Rastegari, M. and Davis, L.S., 2016. G-cnn: an iterative grid based object detector. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2369-2377).
- [24] Redmon, J., Divvala, S., Girshick, R. and Farhadi, A., 2016. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 779-788).
- [25] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y. and Berg, A.C., 2016, October. Ssd: Single shot multibox detector. In *European conference on computer vision* (pp. 21-37). Springer, Cham.
- [26] Redmon, J. and Farhadi, A., 2017. YOLO9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 7263-7271).
- [27] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V. and Rabinovich, A., 2015. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1-9).
- [28] R. Kaushik, S. Kumar, 2019, Image Segmentation Using Convolutional Neural Network, International journal of scientific & technology research volume 8, issue 11, ISSN 2277-8616.
- [29] Ignacio Arganda-Carreras, Srinivas C. Turaga, Daniel R. Berger, Dan Ciresan, Alessandro Giusti, Luca M. Gambardella, Jürgen Schmidhuber, Dmitry Laptev, Sarvesh Dwivedi, Joachim M. Buhmann, Ting Liu, Mojtaba Seyedhosseini, Tolga Tasdizen, Lee Kamentsky, Radim Burget, Vaclav Uher, Xiao Tan, Chanming Sun, Tuan D. Pham, Eran Bas, Mustafa G. Uzunbas, Albert Cardona, Johannes Schindelin, and H. Sebastian Seung. Crowdsourcing the creation of image segmentation algorithms for connectomics. *Frontiers in Neuroanatomy*, vol. 9, no. 142, 2015.
- [30] Ronneberger, O., Fischer, P. and Brox, T., 2015, October. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention* (pp. 234-241). Springer, Cham.
- [31] https://www.cs.umd.edu/class/spring2020/cmsc426-0201/files/18_Homography-estimation-and-decomposition.pdf
- [32] cocodataset.org/
- [33] [cs231n.github.io/](https://github.com/CS231n)

- [34] Sultana, Farhana, Abu Sufian, and Paramartha Dutta. "Advancements in Image Classification Using Convolutional Neural Network." 2018 Fourth International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN), 2018.
- [35] blog.xrds.acm.org/2016/06/convolutional-neural-networks-cnns-illustratedexplanation/
- [36] cs230.stanford.edu/
- [37] topbots.com/semantic-segmentation-guide/
- [38] Ravi Kaushik, Shailender Kumar, Image Segmentation Using Convolutional Neural Network, International journal of scientific & technology research volume 8, issue 11, November 2019
- [39] Tan, M. and Le, Q., 2019, May. Efficientnet: Rethinking model scaling for convolutional neural networks. In International Conference on Machine Learning (pp. 6105-6114). PMLR.
- [40] He, K., Zhang, X., Ren, S. and Sun, J., 2016. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).
- [41] Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M. and Adam, H., 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861.
- [42] Jadon, S., 2020, October. A survey of loss functions for semantic segmentation. In 2020 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB) (pp. 1-7). IEEE.
- [43] github.com/ultralytics/yolov5
- [44] Dubey, S.K. and Vijay, S., 2018. A Review of Image Segmentation using Clustering Methods. Int. J. Appl. Eng. Res, 13, pp.2484-2489.
- [45] Jogin, M., Madhulika, M.S., Divya, G.D., Meghana, R.K. and Apoorva, S., 2018, May. Feature extraction using convolution neural networks (CNN) and deep learning. In 2018 3rd IEEE international conference on recent trends in electronics, information & communication technology (RTEICT) (pp. 2319-2323). IEEE.
- [47] Tuyls, Karl, et al. "Game Plan: What AI can do for Football, and What Football can do for AI." *Journal of Artificial Intelligence Research* 71 (2021): 41-88.

KLJUČNA DOKUMENTACIJSKA INFORMACIJA

Redni broj, RBR :	
Identifikacioni broj, IBR :	
Tip dokumentacije, TD :	monografska publikacija
Tip zapisa, TZ :	tekstualni štampani dokument
Vrsta rada, VR :	diplomski rad
Autor, AU :	Alen Mujo
Mentor, MN :	Dr Jelena Slivka, docent
Naslov rada, NR :	Automatska transformacija snimka utakmice u prikaz iz ptičije perspektive primenom tehnika kompjuterske vizije.
Jezik publikacije, JP :	srpski
Jezik izvoda, Jl :	srpski / engleski
Zemlja publikovanja, ZP :	Srbija
Uže geografsko područje, UGP :	Vojvodina
Godina, GO :	2021
Izdavač, IZ :	autorski reprint
Mesto i adresa, MA :	Novi Sad, Fakultet tehničkih nauka, Trg Dositeja Obradovića 6
Fizički opis rada, FO :	8 / 43 / 32 / 2 / 14 / 0 / 0
Naučna oblast, NO :	Softversko inženjerstvo i informacione tehnologije
Naučna disciplina, ND :	Mašinsko učenje
Predmetna odrednica / ključne reči, PO :	mašinsko učenje, konvolucione neuronske mreže
UDK	
Čuva se, ČU :	Biblioteka Fakulteta tehničkih nauka, Trg Dositeja Obradovića 6, Novi Sad
Važna napomena, VN :	
Izvod, IZ :	Zadatak ovog rada je specifikacija, implementacija i verifikacija sistema za transformaciju snimka utakmice u prikaz iz ptičije perspektive. Rešenje je implementirano treniranjem postojećih modela za segmentaciju ključnih tačaka ulazne slike i modela za detekciju igrača. Za obučavanje modela korišćen je javno dostupan skup podataka koji se sastoji od slika prenosa utakmica.
Datum prihvatanja teme, DP :	
Datum odbrane, DO :	
Članovi komisije, KO :	
predsednik	Dr Aleksandar Kovačević, vanredni profesor
član	Dr Nikola Luburić, docent
mentor	Dr Jelena Slivka, vanredni profesor
Potpis mentora	

KEY WORDS DOCUMENTATION

Accession number, ANO :	
Identification number, INO :	
Document type, DT :	monographic publication
Type of record, TR :	textual material
Contents code, CC :	bachelor thesis
Author, AU :	Alen Mujo
Mentor, MN :	Jelena Slivka, assistant professor, PhD
Title, TI :	Automatic transformation of match footage in a bird's eye view using computer vision techniques.
Language of text, LT :	Serbian
Language of abstract, LA :	Serbian / English
Country of publication, CP :	Serbia
Locality of publication, LP :	Vojvodina
Publication year, PY :	2021
Publisher, PB :	author's reprint
Publication place, PP :	Novi Sad, Faculty of Technical Sciences, Trg Dositeja Obradovića 6
Physical description, PD :	8 / 43 / 32 / 2 / 14 / 0 / 0
Scientific field, SF :	Software Engineering and Information Technologies
Scientific discipline, SD :	Machine Learning
Subject / Keywords, S/KW :	machine learning, convolutional neural networks
UDC	
Holding data, HD :	Library of the Faculty of Technical Sciences, Trg Dositeja Obradovića 6, Novi Sad
Note, N :	
Abstract, AB :	The task of this paper is the specification, implementation and verification of the system for the transformation of recorded matches into a bird's eye view. The solution is implemented by training existing models for segmentation of key points of input images and models for player detection. A publicly available data set consisting of a match broadcast image was used to train the model.
Accepted by sci. Board on, ASB :	
Defended on, DE :	
Defense board, DB :	
president	Aleksandar Kovačević, associate professor, PhD
member	Nikola Luburić, assistant professor, PhD
mentor	Jelena Slivka, associate professor, PhD
Mentor's signature	