Sustav za detekciju i identifikaciju većeg broja proizvoda putem QR-kôda

Rašić, Marin

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 321

SUSTAV ZA DETEKCIJU I IDENTIFIKACIJU VEĆEG BROJA PROIZVODA PUTEM QR-KÔDA

Marin Rašić

Zagreb, veljača 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 321

SUSTAV ZA DETEKCIJU I IDENTIFIKACIJU VEĆEG BROJA PROIZVODA PUTEM QR-KÔDA

Marin Rašić

Zagreb, veljača 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Zagreb, 2. listopada 2023.

DIPLOMSKI ZADATAK br. 321

Pristupnik:	**Marin Rašić (0036514528)**
Studij:	Računarstvo
Profil:	Računalno inženjerstvo
Mentor:	izv. prof. dr. sc. Daniel Hofman

Zadatak: **Sustav za detekciju i identifikaciju većeg broja proizvoda putem QR-kôda**

Opis zadatka:

Proučiti rad sustava za detekciju QR-kôdova i sustava koji koriste umjetnu inteligenciju za detekciju objekata. Odabrati model za detekciju objekata. Dodatno optimizirati model kako bi kvalitetnije detektirao objekte. Implementirati prepoznavanje i čitanje podataka iz QR-kôdova na detektiranim objektima. Izmjeriti performanse sustava kao što su brzina i kvaliteta prepoznavanja. Optimizirati sustav kako bi radio s više od 10 objekata u realnom vremenu. Izraditi aplikaciju za detektiranje objekata i QR-kôdova. Napraviti dokumentaciju i objaviti izvorni kôd programskog rješenja na repozitoriju Git.

Rok za predaju rada: 9. veljače 2024.

Sadržaj

1. Uvod

Ljudima je dovoljan jedan pogled na sliku i odmah mogu zaključiti koji se objekti nalaze na slici, gdje se nalaze i u kakvom su odnosu s drugim objektima na slici. Ljudski vizualni sustav je brz i efikasan što nam omogućuje obavljanje vrlo kompleksnih zadataka poput vožnje automobila ili igranje sporta. I dok je ljudima detekcija objekata trivijalna stvar, u području računalnog vida to je jedan od najznačajnijih problema.

Detekcija objekata se primjenjuje u raznim područjima na različite načine. Koristi se u samovozećim automobilima za detekciju cesta, pješaka, prometnih znakova i drugih vozila, za skeniranje lica na aerodromima, u zdravstvu se koristi za pomoć u dijagnostici, u poljoprivredi za pregled stanja tla i praćenje rasta i zdravlja biljaka, itd. Moderni modeli detekcije objekata temelje se na dubokom učenju i konvolucijskim neuralnim mrežama. Razlikujemo dvije glavne grupe modela: dvofazni koji procesiraju sliku dvaput i jednofazni koji detekciju obavljaju u jednom prolazu. Općenito su dvofazni modeli točniji ali sporiji, dok su jednofazni modeli manje točni ali su dovoljno brzi da omoguće detekciju objekata u stvarnom vremenu.

Detekcija QR kodova je još jedno važno područje računalnog vida. QR kodovi su matrice crno-bijelih ćelija koji kodiraju nekakvu informaciju. Koriste se za praćenje pošiljka, pristup internetskim stranicama, za lagano dijeljenje informacija za plaćanje i na još mnoge druge načine.

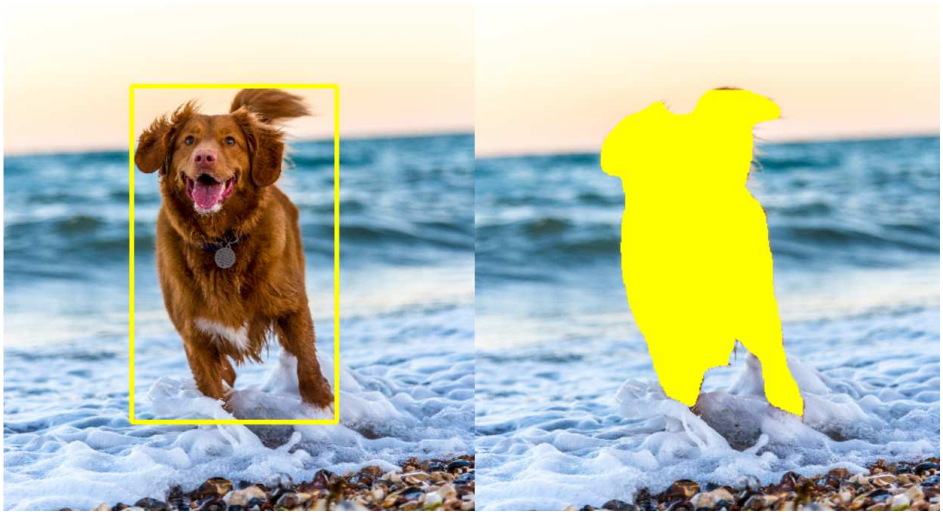
U sklopu ovog rada implementiran je kod koji kombinira detekciju objekata te detekciju i dekodiranje QR kodova na njima. Konkretno, provodi se detekcija kutija kao npr. pošiljka te prikaz informacija o QR kodu koji se nalaze na njima. Željela se postići detekcija u stvarnom vremenu većeg broja kutija odjednom.

2. Detekcija objekata

Detekcija objekata je tehnika računalnog vida čiji je cilj pronalaženje objekata na slici. To je jedan od ključnih i najzahtjevnijih problema računalnog vida (Zou, 2019). Objekti se na slici mogu pojaviti na različitim lokacijama, u različitom broju te u različitim veličinama. Što neki objekt predstavlja nazivamo klasa objekta, npr. automobil, kutija, jabuka, ljudsko lice i slično.

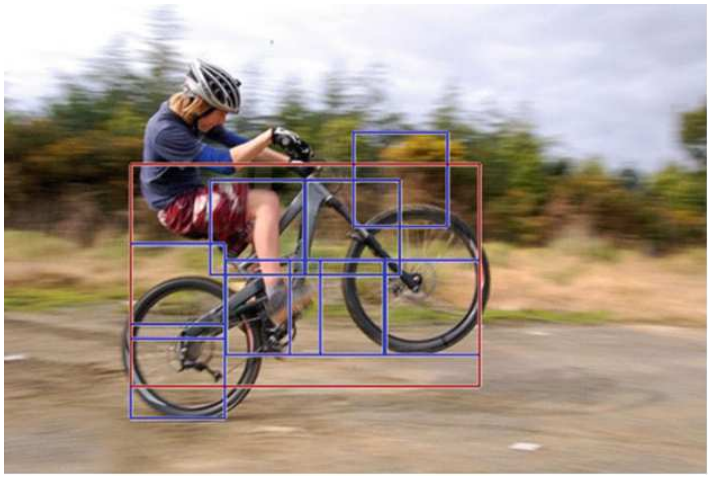
Problem detekcije objekata općenito možemo rastaviti u tri glavna zadataka: određivanje postoji li objekt na slici, određivanje gdje se taj objekt (ako postoji) nalazi na slici, te određivanje što je taj objekt tj. određivanje klase objekta (Kumari, 2023).

Ulaz u detektor objekata je slika, a svaka detekcija rezultira nekakvom informacijom. Ta informacija može biti jednostavna: lokacija objekta na slici, njegova veličina, objektov granični okvir (engl. *bounding box*) ili objektova segmentacijska maska (engl. *segmentation mask*). Granični okvir je pravokutnik koji sadrži objekt, a segmentacijska maska je obris objekta. Slika 2.1. prikazuje primjer graničnog okvira i segmentacijske maske.



Slika 2.1. Primjer graničnog okvira i segmentacijske maske

Informacija dobivena detekcijom može biti i kompleksna te sadržavati parametre linearne ili nelinearne transformacije (Amit, 2021). Na primjer, slika 2.2. prikazuje detektor bicikla koji nakon što detektira bicikl može odrediti i pojedine dijelove bicikla.



Slika 2.2. Primjer kompleksne informacije dobivene detekcijom objekata (Amit, 2021)

Rezultat detekcije objekata općenito možemo zamisliti kao vektor vrijednosti:

$$\begin{bmatrix} x_1 & y_1 & x_2 & y_2 & p & c_1 & c_2 & \dots & c_n \end{bmatrix} \tag{2.1}$$

gdje vrijednosti x_1, y_1, x_2 i y_2 predstavljaju koordinate graničnog okvira unutar kojeg se objekt moguće nalazi: točka $(x1, y1)$ predstavlja gornji lijevi kut graničnog okvira, a točka $(x2, y2)$ donji desni kut. Vrijednost p predstavlja vjerojatnost da se objekt nalazi unutar tog graničnog okvira. Vrijednosti $c_1, c_2, \dots, c_n$ govore kojoj klasi pripada objekt. Ako je $c_x = 1$ tada objekt pripada klasi x. Samo jedna od c vrijednosti može biti jednaka jedan, ostale će uvijek biti jednake nula (Kumari, 2023).

Slika 2.3. Primjer rada detektora objekata

2.1. Razvoj detekcije objekata

Razvoj tehnologije detekcije objekata započeo je devedesetih godina 20. stoljeća. Razvoj možemo grubo podijeliti u dvije faze: faza tradicionalne detekcije objekata (do 2014. godine) i faza detekcije objekata bazirana na dubokom učenju (od 2014. godine) (Zou, 2019).

U fazi tradicionalne detekcije objekata većina detektora je bila izrađena za točno određene zadatke. Algoritmi su bili specifični za problem koji su rješavali i nisu se mogli ponovno iskoristiti za druge projekte. Jedan od prvih značajnijih algoritama ove faze je algoritam za prepoznavanje ljudskog lica u stvarnom vremenu kojeg su 2001. godine objavili P. Viola i M. Jones (Zou, 2019). Ovaj algoritam se temeljio na konceptu klizećeg prozora (engl. *sliding window*). Prozor bi "klizio" kroz sliku i u svakom koraku provjeravao nalazi li se na tom dijelu slike ljudsko lice.

Razvoj detekcije objekata počeo je stagnirati oko 2010. godine i tražilo se neko novo rješenje. Revoluciju u području detekcije objekata donijele su neuronske mreže i duboko učenje. R. Girshick, J. Donahue, T. Darrell i J. Malik su 2014. godine na CPVR (Computer Vision and Pattern Recognition) konferenciji predstavili ideju detekcije objekata koristeći konvolucijsku neuronsku mrežu. Time je započela druga faza razvoja detekcije objekata bazirana na dubokom učenju (Zou, 2019).

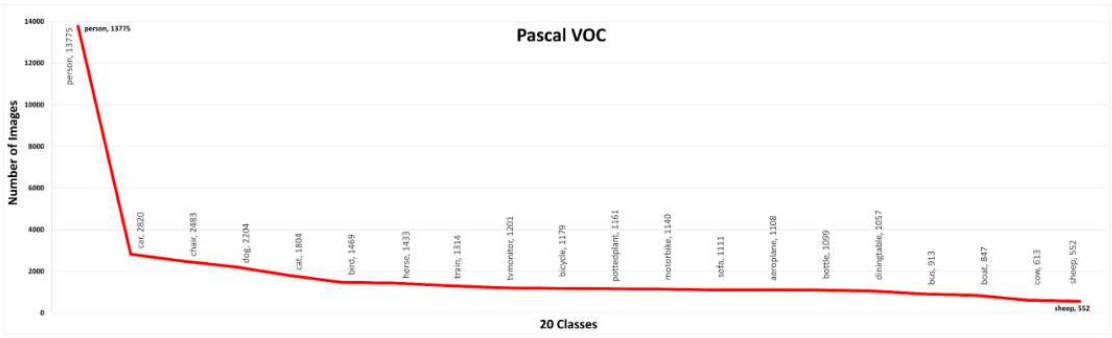
U drugoj fazi, razvoj tehnologije detekcije objekata se značajno ubrzao zahvaljujući brzom razvoju tehnika dubokog učenja i puno većoj zainteresiranost znanstvene zajednice. To pokazuje i činjenica da je broj publikacija na temu detekcije objekata znatno porastao u drugoj fazi razvoja. Broj publikacija na temu detekcije objekata 2014. godine bio je oko 750, a 2021. godine bilo je gotovo 3500 publikacija (Zou, 2019). U ovoj fazi razlikujemo dvije skupine detektora: dvofazni (engl. *two-stage*) i jednofazni (engl. *one-stage*) detektori. Dvofazni detektori rade na principu grube i fine detekcije, dok jednofazni rade na principu detekcije unutar jednog koraka (Zou, 2019).

Prvi predloženi jednofazni detektor je bio YOLO (You Only Look Once). Predstavili su ga 2015. godine J. Redmon, S. Divvala, R. Girshick i A. Farhadi u svom radu You Only Look Once: Unified, Real-Time Object Detection (Redmon, 2016).

2.2. Skupovi podataka za detekciju objekata

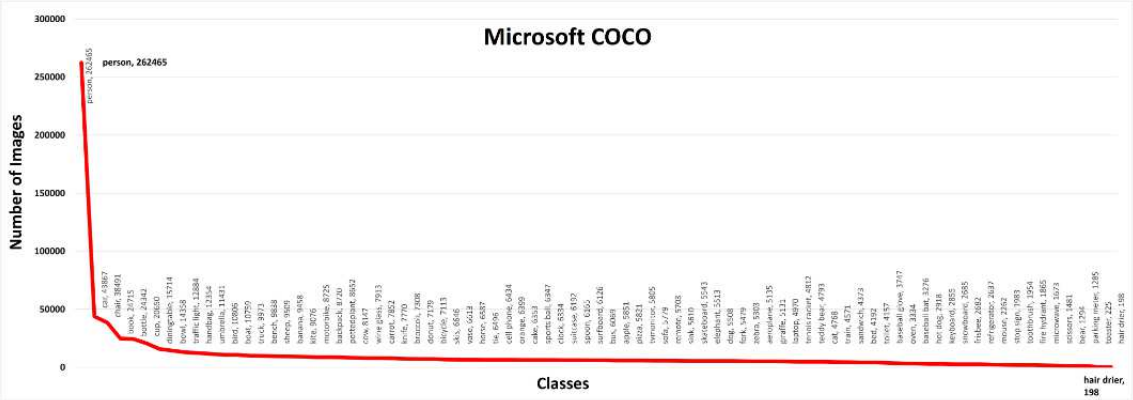
Detektori objekata konstruiraju model ("uče") na temelju skupa primjera za treniranje. Podatke koji se koriste za treniranje modela možemo opisati kao parove slika i vektora vrijednosti. Svaki objekt na slici je uparen s jednim vektorom vrijednosti. Taj vektor vrijednosti možemo zamisliti jednako kao i vektora vrijednosti rezultat detekcije 2.1 Vjerojatnost p će u podatcima za treniranje uvijek biti jednaka 1 zato što znamo da se objekt sigurno nalazi unutar graničnog okvira.

U svrhu treniranja modela detekcije objekata nastali su brojni veliki skupovi podataka. Prvi poznatiji od takvih skupova podataka je *PASCAL Visual Object Classes (VOC)*. Rad na njemu je započeo 2005. godina s ciljem ubrzavanja razvoja detekcije objekata (Zaidi, 2021). Postoji više verzija *PASCAL VOC* skupa podataka, ali najpopularnije su *VOC07* i *VOC12*. Ove verzije se i danas koriste za treniranje i testiranje modela detekcije objekata. Verzija *VOC07* imala je četiri klase objekata i sastojala se od pet tisuća slika s preko 12 tisuća označenih objekata na njima. Verzija *VOC12* povećala je broj klasa s četiri na 20, te je imala 11 tisuća slika s više od 27 tisuća označenih objekata.



Slika 2.4. Broj slika različitih klasa objekata u PASCAL VOC12 skupu podataka (Zaidi, 2021)

Microsoft Common Objects in Context (MS-COCO) je još jedan popularni skup podataka. Projekt izrade *MS-COCO*-a započeo je 2015. godine. Ideja iza *MS-COCO*-a je bila napraviti skup podataka koji sadrži česte objekte u njihovom normalnom okruženju koje četverogodišnje dijete može prepoznati (Zaidi, 2021). Sadrži preko dva milijuna slika na kojima se može pronaći 91 različitih klasa objekata.



Slika 2.5. Broj slika različitih klasa objekata u MS-COCO skupu podataka (Zaidi, 2021)

Uz *PASCAL VOC* i *MS-COCO* postoje još brojni drugi skupovi podataka dizajnirani za detekciju objekata poput *ImageNet Large Scale Visual Recognition Challenge (ILSVRC)* i *OpenImage*. Mnogi od njih su kreirani za točno određenu ulogu npr. *Karlsruhe Institute of Technology and Toyota Technological Institute (KITTI)* je vrlo popularan skup podataka za detekciju objekata u području autonomne vožnje.

Dataset	Classes	Train			Validation			Test
		Images	Objects	Objects/Image	Images	Objects	Objects/Image	
PASCAL VOC 12	20	5,717	13,609	2.38	5,823	13,841	2.37	10,991
MS-COCO	80	118,287	860,001	7.27	5,000	36,781	7.35	40,670
ILSVRC	200	456,567	478,807	1.05	20,121	55,501	2.76	40,152
OpenImage	600	1,743,042	14,610,229	8.38	41,620	204,621	4.92	125,436

Slika 2.6. Usporedba različitih skupova podataka za detekciju objekata (Zaidi, 2021)

2.3. Izazovi u detekciji objekata

Varijacije među objektima istih klasa jedan od najznačajnijih problema detekcije objekata. Potrebno je dizajnirati algoritam detekcije objekata koji je invarijantan na te različitosti (Amit, 2021). Najjednostavniji način rješavanja ovog problema je gruba sila. Koriste se ogromne količine primjera za treniranje kako bi se uhvatile moguće varijacije. Primjeri takvih skupova podataka dani su u poglavlju 2.2. Ako je količina primjera za treniranje limitirana potrebno je invarijantnost ugraditi u model. Jedna od metoda za ugrađivanje invarijantnosti je računanje invarijantnih funkcija i značajka, a druga obuhvaća pretraživanje latentnih varijabla (Amit, 2021). Najčešće se ove dvije metode kombiniraju.

Veliki izazov s kojim se područje detekcije objekata susreće je efikasnost. Današnji modeli zahtijevaju velike računalne resurse kako bi ostvarili prihvatljivu točnost (Zaidi, 2021). S velikim porastom korištenja mobilnih uređaja efikasnost postaje sve bitnija.

Nedostatak visoko kvalitetnih skupova podataka je još jedan problem. Iako postoji velik broj skupova podataka opet ih nema dovoljno i nisu svi dovoljno kvalitetni (Zaidi, 2021). Čak i široko korištenih skupovi podataka poput onih opisanih u poglavlju 2.2. imaju neke svoje probleme. Primjerice, ako pogledamo graf 2.5. možemo vidjeti da *MS-COCO* skup podataka sadrži 262465 slika s klasom "čovjek", a samo 198 slika za klasu "sušilo za kosu". Ovakva velika razlika će nužno rezultirati problemima pristranosti u treniranju bilo kojeg modela detekcije objekata (Zaidi, 2021).

Postoje još brojne druge i raznovrsne prepreke i izazovi s kojim se susreće u detekciji objekata. Neki od tih problema su ogroman broj različitih klasa objekata, velik broj objekata na slici, raznovrsnost objekata na slici, različito osvjetljenje objekata, različite točke gledišta na isti objekt, različite veličine objekata i njihova rotacija, samo djelomično vidljivi objekti, itd. (Zou, 2019).

2.4. Metrike detekcije objekata

Dvije najvažnije metrike detekcije objekata su točnost i brzina. Točnost uključuje točnu klasifikaciju objekta i točnu lokalizaciju objekta tj. točno određivanje gdje se objekt nalazi na slici (Zou, 2019). Točnost i brzina su negativno korelirane tj. kako bi model detekcije postigao veću brzinu mora žrtvovati točnost i obratno.

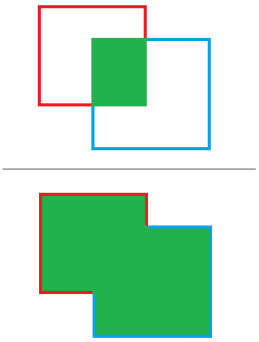
Prema brzini, modele detekcije objekata dijelimo na one koji detekciju mogu provesti u stvarnom vremenu i one koji ne mogu. Jedna od mjera brzine je broj slika u sekundi koje sustav može obraditi ili FPS (Frames per second).

Jedan od načina mjerenja točnosti detekcije objekata je Presjek preko Unije (engl. *Intersection over Union*) ili Jaccardov indeks, skraćeno *IoU*. *IoU* daje mjeru preklapanje stvarnog graničnog okvira (engl. *ground truth*) i graničnog okvira dobivenog detekcijom objekata.

Formula za računanje IoU-a je sljedeća:

$$IoU = \frac{P_i}{P_u} \tag{2.2}$$

gdje je P_i površina presjeka graničnih okvira, a P_u je površina unije graničnih okvira (Rosebrock, 2016). IoU će uvijek imat vrijednost u segmentu $[0, 1]$. Što je ta vrijednost veća to se granični okviri više podudaraju tj. veća je točnost detekcije.

$IoU =$  (2.3)

Još dvije popularne metrike mjerenja točnosti detekcije objekata su prosječna preciznost (engl. *average precision*) ili *AP* i srednja prosječna preciznost (engl. *mean average precision*) ili *mAP*. Kako bi definirali *AP* i *mAP* prvo ćemo definirati još dva pojma: preciznost (engl. *precision*) i *recall*.

Preciznost je mogućnost modela da detektira samo relevantne objekte. To je postotak točnih pozitivnih predikcija tj. omjer točnih pozitivnih detekcija i svih detekcija. *Recall* predstavlja mogućnost modela da pronađe sve relevantne objekte. To je omjer točnih detekcija i svih mogućih točnih detekcija (Padilla, 2021). Preciznost i *recall* će uvijek imati vrijednosti unutar segmenta $[0, 1]$ te ih definiramo sljedećim formulama:

$$precision = \frac{TP}{TP + FP} \tag{2.4}$$

$$recall = \frac{TP}{TP + FN} \tag{2.5}$$

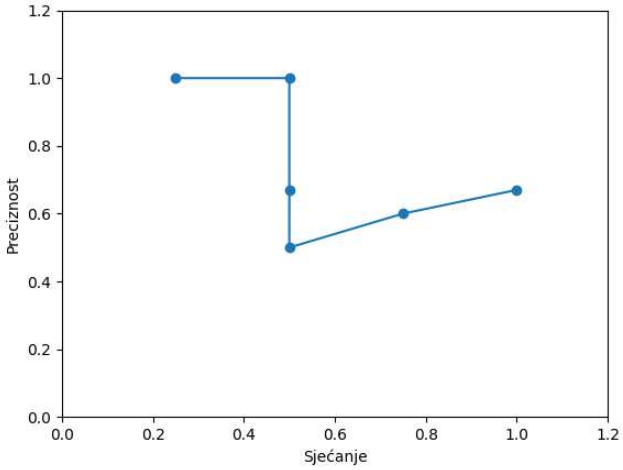
gdje je TP ukupan broj ispravnih pozitivnih detekcija, FP je ukupan broj pogrešnih pozitivnih detekcija, a FN je ukupan broj pogrešnih negativnih detekcija.

Na temelju vrijednosti preciznosti i *recall*-a možemo stvoriti *precision-recall* krivulju (Hui, 2018). Stvaranje *precision-recall* krivulje demonstrirat ćemo na primjeru. Pretpostavimo da imamo skup podataka na kojima provodimo detekciju jedne klase npr. kutije te da se na slikama unutar skupa podataka nalaze sveukupno četiri kutije tj. $TP + FN = 4$. Provedemo detekciju objekata i popišemo sve predikcije te ih sortiramo padajuće prema vrijednost p vektora rezultata detekcije 2.1 . Tablica 2.1. prikazuje moguće rezultate preciznosti i *recall*-a za dani primjer.

Tablica 2.1. *Precision-recall* tablica

	Correct detection	Precision	Recall
1.	✓	1.0	0.25
2.	✓	1.0	0.5
3.	✗	0.67	0.5
4.	✗	0.5	0.5
5.	✓	0.6	0.75
6.	✓	0.67	1.0

Ako dobivene vrijednosti preciznosti i sjećanja nacrtamo u dvodimenzionalnom koordinatnom sustavu dobit ćemo *precision-recall* krivulju (Hui, 2018). Vrijednosti *recall*-a ćemo postaviti na apscisu, a preciznosti na ordinatu. Slika 2.7. prikazuje *precision-recall* krivulju dobivenu za tablicu 2.1.



Slika 2.7. *Precision-recall* krivulja

Prosječnu preciznost ili *AP* definiramo kao površinu ispod *precision-recall* krivulje:

$$AP = \int_0^1 p(r)dr \tag{2.6}$$

AP se definira za samo jednu klasu objekata. Srednja prosječna preciznost ili *mAP* je srednja vrijednost svih prosječnih preciznost za različite klase objekata. Formula za *mAP* je sljedeća:

$$mAP = \frac{1}{n}\sum_{i=1}^{i=n} AP_i \tag{2.7}$$

gdje je n broj različitih klasa objekata, a AP_i je prosječna preciznost za klasu i (Gad, 2021).

	backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Two-stage methods							
Faster R-CNN+++ [5]	ResNet-101-C4	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w FPN [8]	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN by G-RMI [6]	Inception-ResNet-v2 [21]	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w TDM [20]	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	**52.1**
One-stage methods							
YOLOv2 [15]	DarkNet-19 [15]	21.6	44.0	19.2	5.0	22.4	35.5
SSD513 [11, 3]	ResNet-101-SSD	31.2	50.4	33.3	10.2	34.5	49.8
DSSD513 [3]	ResNet-101-DSSD	33.2	53.3	35.2	13.0	35.4	51.1
RetinaNet [9]	ResNet-101-FPN	39.1	59.1	42.3	21.8	42.7	50.2
RetinaNet [9]	ResNeXt-101-FPN	**40.8**	**61.1**	**44.1**	**24.1**	**44.2**	51.2
YOLOv3 608 × 608	Darknet-53	33.0	57.9	34.4	18.3	35.4	41.9

Slika 2.8. Usporedba prosječne preciznosti različitih modela detekcije objekata (Redmon, 2018)

Model	Year	Backbone	Size	$AP_{[0.5:0.95]}$	$AP_{0.5}$	FPS
R-CNN*	2014	AlexNet	224	-	58.50%	~0.02
SPP-Net*	2015	ZF-5	Variable	-	59.20%	~0.23
Fast R-CNN*	2015	VGG-16	Variable	-	65.70%	~0.43
Faster R-CNN*	2016	VGG-16	600	-	67.00%	5
R-FCN	2016	ResNet-101	600	31.50%	53.20%	~3
FPN	2017	ResNet-101	800	36.20%	59.10%	5
Mask R-CNN	2018	ResNeXt-101-FPN	800	39.80%	62.30%	5
DetectoRS	2020	ResNeXt-101	1333	53.30%	71.60%	~4
YOLO*	2015	(Modified) GoogLeNet	448	-	57.90%	45
SSD	2016	VGG-16	300	23.20%	41.20%	46
YOLOv2	2016	DarkNet-19	352	21.60%	44.00%	81
RetinaNet	2018	ResNet-101-FPN	400	31.90%	49.50%	12
YOLOv3	2018	DarkNet-53	320	28.20%	51.50%	45
CenterNet	2019	Hourglass-104	512	42.10%	61.10%	7.8
EfficientDet-D2	2020	Efficient-B2	768	43.00%	62.30%	41.7
YOLOv4	2020	CSPDarkNet-53	512	43.00%	64.90%	31
Swin-L	2021	HTC++	-	57.70%	-	-

[a]Models marked with * are compared on PASCAL VOC 2012, while others on MS COCO.Rows colored gray are real-time detectors (>30 FPS).

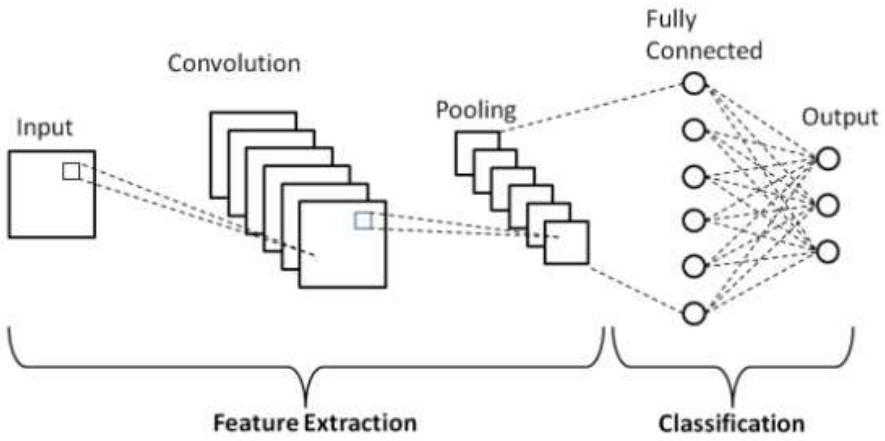
Slika 2.9. Usporedba točnosti i brzine različitih modela detekcije objekata (Zaidi, 2021)

2.5. Konvolucijske neuronske mreže

Konvolucijska neuronska mreža ili CNN (Convolutional Neural Network) je klasa neuronskih mreža specijalizirana u procesiranju podataka koji se mogu podijeliti u manje dijelove - ćelije (Mishra, 2020). CNN-ovi se najčešće koriste za obradu slika koje možemo zamisliti kao matricu ćelija gdje svaka ćelija sadrži jedan piksel.

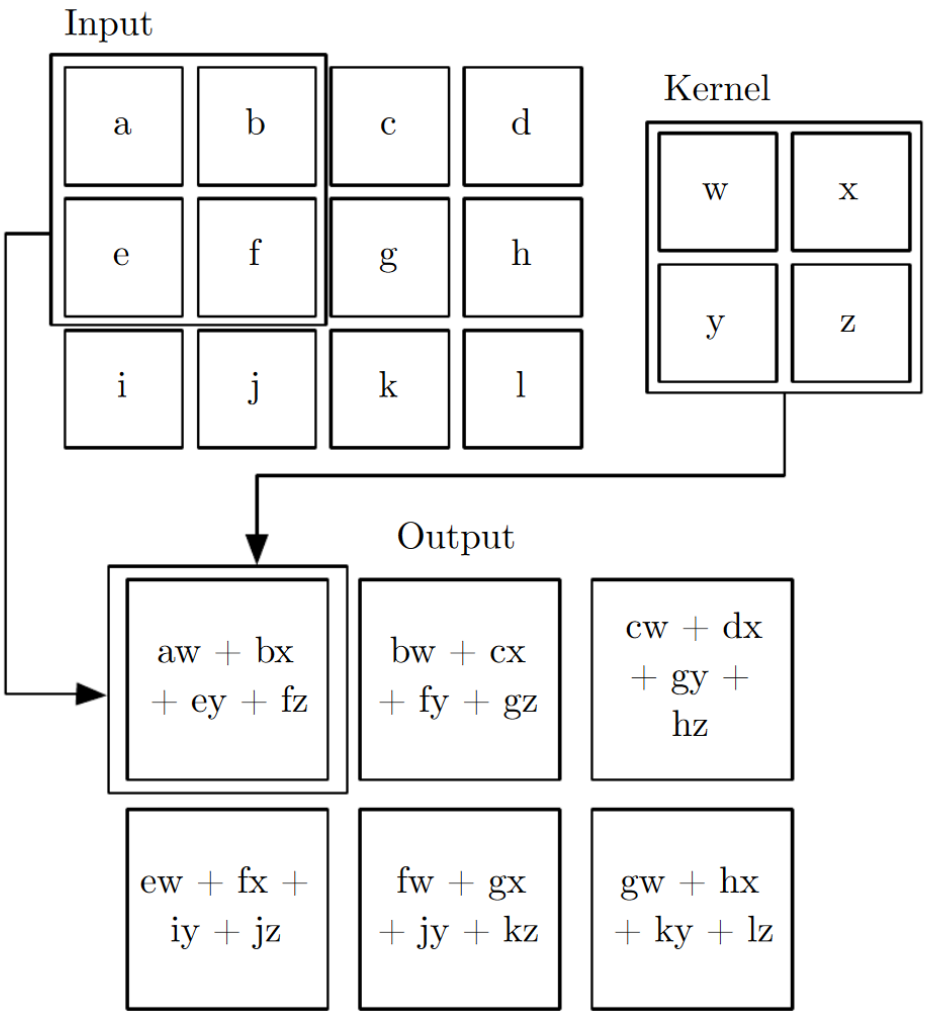
Konvolucijske neuralne mreže inspirirane su ljudskim vizualnim sustavom. Pogledom na sliku u ljudskom mozgu se aktiviraju brojni neuroni. Svaki od tih neurona odgovoran je za točno određeni dio vidnog polja i povezan je s ostalim neuronima na takav način da zajedno prekrivaju čitavo vidno polje. Zbog toga i neuroni u konvolucijskim neuronskim mrežama procesiraju samo određeni dio slike (Mishra, 2020).

Općenito, CNN se sastoji od tri vrste slojeva: konvolucijski slojevi (engl. *convolutional layers*), slojevi sažimanja (engl. *pooling layers*) i potpuno povezani slojevi (engl. *fully connected layers*) (Mishra, 2020). Slojevi su poredani na takav način da se prvo detektiraju jednostavni uzorci na slici poput točaka, linija i krivulja, a kasnije kompleksni poput objekata ili lica.



Slika 2.10. Arhitektura konvolucijske neuralne mreže (Phung, 2019)

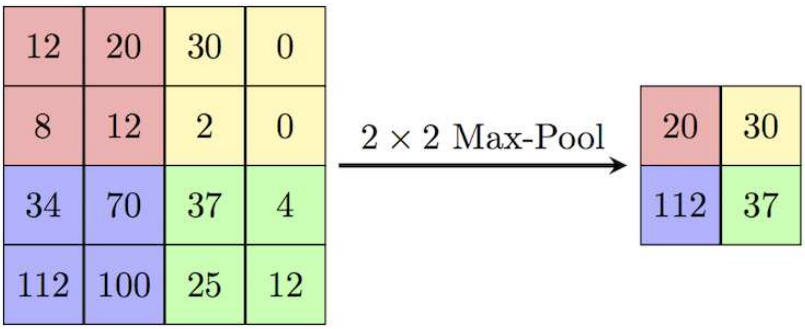
Konvolucijski slojevi su srž konvolucijskih neuronskih mreža te se većina računanja u mreži događa unutar njih. Oni provode matrično množenje dvije matrice: matrice parametre učenja ili kernela i matrice određenog dijela slike (Mishra, 2020). Kernel se množi sa svakim pojedinim dijelom slike i skup rezultata množenja nazivamo aktivacijska mapa (engl. *activation map*).



Slika 2.11. Aktivacijska mapa (Bengio, 2017)

Tri glavne ideje iza korištenja konvolucije su slaba interakcija, dijeljenje parametra i ekvivarijantna reprezentacija (Mishra, 2020). Kernel je znatno manji od cijele slike i on se množi samo s pojedinim dijelovima slike. Time se ostvaruje ideja slabe interakcije ali i ideja dijeljenja parametra. Različiti dijelovi slike se neovisno jedan o drugom množe s kernelom, ali se prilikom množenja koriste isti parametri iz kernela (Mishra, 2020).

Slojevi sažimanja služe kako bi se smanjila količina podataka s kojima mreža mora raditi. Oni smanjuju izlaze prethodnih slojeva i time reduciraju potrebnu količinu računanja u daljnjim slojevima. Postoje mnoge funkcije za sažimanje poput prosjek skupa pravokutnika, L2 norma skupa pravokutnika, itd. Najpopularnija metoda je maksimum skupa pravokutnika koja kao ulaz uzima skup pravokutnika, a kao izlaz daje najveću vrijednost iz skupa pravokutnika (Mishra, 2020).



Slika 2.12. Maksimum skupa pravokutnika (engl. *max pooling*) (computersciencewiki.org)

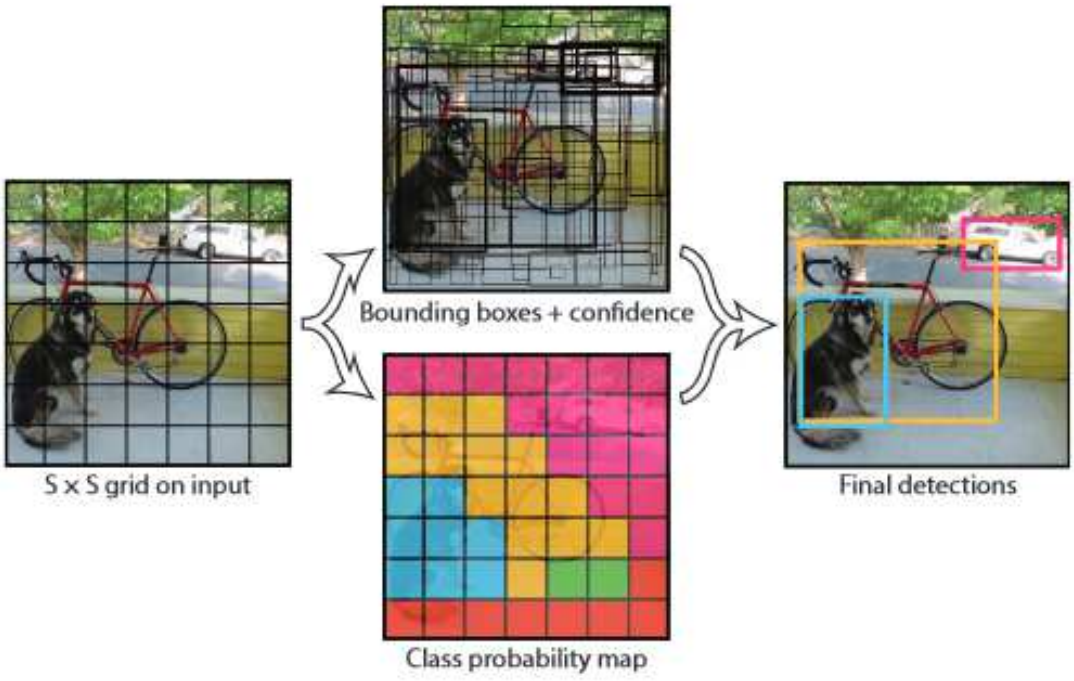
Potpuno povezani slojevi sastoje se od neurona koji su u potpunosti povezani sa svim neuronima u prethodnom i sljedećem sloju (Mishra, 2020). Ovi slojevi se najčešće koriste kako bi dvodimenzionalne podatke iz prethodnih slojeva pretvorili u jednodimenzionalni vektor. U detekciji objekata ovaj dio mreže je zadužen za klasifikaciju objekata (Rastogi, 2023).

2.6. YOLO model

You Only Look Once (YOLO) je sustav za detekciju objekata u stvarnom vremenu. Kao što mu i ime sugerira, YOLO je jednofazni sustav detekcije objekata tj. detekcija se obavlja u jednom koraku. YOLO unificira odvojene komponente dotadašnjih sustava za detekciju u jednu neuralnu mrežu (Redmon, 2016). YOLO je isprva bio implementiran koristeći *Darknet framework*.

U YOLO modelu svaki granični okvir se određuje na temelju cijele slike te se svi granični okvire za sve klase određuju istovremeno. To znači da neuronska mreža uzima u obzir podatke o cijeloj slici i svim objektima na njoj (Redmon, 2016).

YOLO model prvo dijeli sliku u SxS jednako velikih ćelija. Svaka ćelija je odgovorna za detekciju svih objekata čiji centar upada unutar te ćelije te svaka ćelija predviđa B graničnih okvira i vrijednost pouzdanosti (engl. *confidence score*) za te okvire (Redmon, 2016). Vrijednost pouzdanosti govori koliko je model siguran da se objekt nalazi unutar graničnog okvira i da je granični okvir točan.



Slika 2.13. YOLO Model (Redmon, 2016)

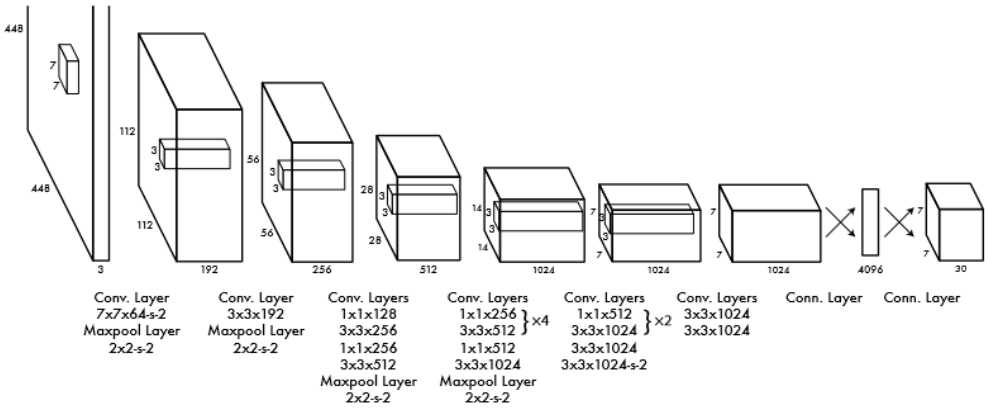
Kao i za općeniti sustav detekcije objekata i kod YOLO modela rezultat možemo zamisliti kao vektor vrijednosti:

$$\begin{bmatrix} x & y & w & h & conf & c_1 & c_2 & \dots & c_n \end{bmatrix} \tag{2.8}$$

gdje vrijednosti x i y predstavljaju koordinate centra graničnog okvira relativno na granice ćelije, a vrijednosti w i h predstavljaju visinu i širinu graničnog okvira. Vrijednost $conf$ je vrijednost pouzdanosti. Vrijednosti $c_1, c_2, \dots, c_n$ predstavljaju vjerojatnosti da objekt pripada nekoj klasi. Za svaku ćeliju se računa samo jedan set vrijednosti $c_1, c_2, \dots, c_n$ bez obzira na broj graničnih okvira čiji se centar nalazi u toj ćeliji i za koje je ta ćelija odgovorna. Množenjem vrijednosti c_x s vrijednošću pouzdanosti dobivamo vrijednost pouzdanosti specifičan za klasu x.

$$c_x * conf = conf_x \tag{2.9}$$

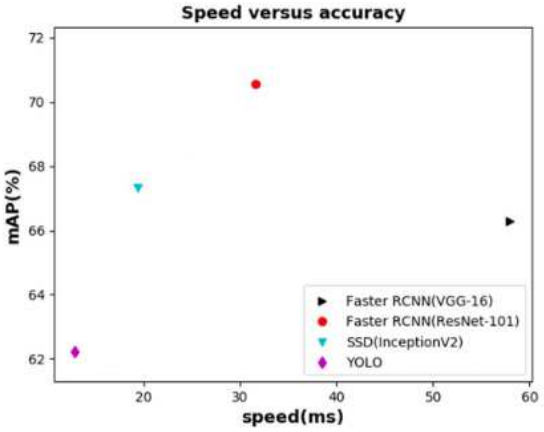
Neuronska mreža korištena u YOLO modelu sastoji se od 24 konvolucijska sloja i dva potpuno povezana sloja (Redmon, 2016). Koriste se $1x1$ slojevi sažimanja praćeni $3x3$ konvolucijskim slojevima. Slika 2.14. prikazuje arhitekturu YOLO neuronske mreže. Rezultat mreže je $7 * 7 * 30$ tenzor predikcija (Redmon, 2016). Postoji još jedna varijanta arhitekture neuronske mreže koja se sastoji od samo devet konvolucijskih slojeva. Ova verzija arhitektura osmišljena je za brzu verziju YOLO modela nazvana Fast YOLO.



Slika 2.14. Arhitektura YOLO neuronske mreže (Redmon, 2016)

Kroz godine je proizvedeno niz novih verzija YOLO modela: YOLOv1, YOLOv2, YOLOv3, YOLOv4, YOLOv5, YOLOv6, YOLOv7 i najnoviji YOLOv8. Svaka verzija je izgrađena na temelju prethodne i na neki način ju poboljšava.

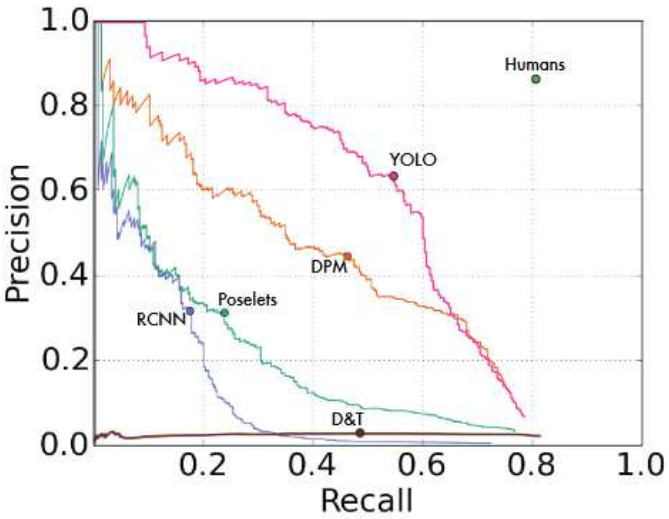
Najveća prednost YOLO sustava nad drugim sustavima detekcije objekata je njegova brzina koja omogućuje detekciju objekata u stvarnom vremenu. YOLO zahtjeva minimalno podataka za treniranje te ima vrlo jednostavnu arhitekturu što omogućuje laganu implementaciju i korištenje YOLO modela u različitim projektima. Glavni nedostatak YOLO modela je loša detekciji grupa sitnih objekata (Redmon, 2016). Zbog činjenice da YOLO model predikciju graničnih okvira uči direktno iz podataka, model ima probleme s detekcijom objekata u novim ili neobičajnim proporcijama.



Slika 2.15. Usporedba YOLO modela i drugih modela detekcije objekata (Yadav, 2017)

Real-Time Detectors	Train	mAP	FPS
100Hz DPM [30]	2007	16.0	100
30Hz DPM [30]	2007	26.1	30
Fast YOLO	2007+2012	52.7	**155**
YOLO	2007+2012	**63.4**	45
Less Than Real-Time			
Fastest DPM [37]	2007	30.4	15
R-CNN Minus R [20]	2007	53.5	6
Fast R-CNN [14]	2007+2012	70.0	0.5
Faster R-CNN VGG-16[27]	2007+2012	73.2	7
Faster R-CNN ZF [27]	2007+2012	62.1	18
YOLO VGG-16	2007+2012	66.4	21

Slika 2.16. Usporedba YOLO modela i drugih modela detekcije objekata treniranih na *PASCAL VOC07/12* skupu podataka (Redmon, 2016)



Slika 2.17. Usporedba YOLO modela i drugih modela detekcije objekata nad *Picasso* skupu podataka (Redmon, 2016)

3. QR kod

QR (Quick Response) kod je dvodimenzionalni simbol koji može pohranjivati nekakvu informaciju. Kodirana informacija može biti tekst, URL ili neki drugi podatak. Razvila ga je 1994. godine tvrtka Denso, član Toyota grupe. Originalno namijenjeni za korištenje u proizvodnji automobilskih dijelova, QR kodovi su danas široko korišteni u mnogim područjima (Soon, 2008). Koriste se za praćenje pošiljaka, označavanje proizvoda, za pristup internetskim stranicama i slično.

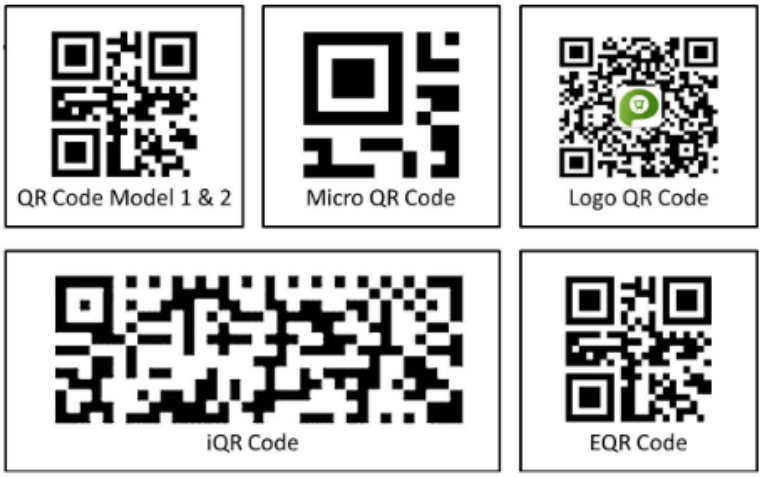
QR kod sustav je vrlo jednostavan, sastoji se od enkodera i dekodera. Enkoder kodira informaciju i generira QR kod koji sadrži tu informaciju, a dekoder dekodira informaciju pohranjenu u QR kodu (Soon, 2008).

Slika 3.1. QR kod sustav (Tiwari, 2016)

Jedan od razloga razvitka QR koda je bila ograničenost tada korištenih bar kodova koji mogu sadržavati maksimalno 20 alfanumeričkih znakova. Kapacitet QR koda znatno je veći, QR kod može sadržavati do 7089 brojeva ili 4296 alfanumeričkih znakova (Tiwari, 2016). Dodatno, QR kod podržava enkodiranje kineskih znakova (Soon, 2008).

Postoje mnoge verzije QR koda, ali općenito ih možemo podijeliti u pet kategorija: Model 1 i 2 QR kod, Micro QR kod, Logo QR kod, iQR kod i EQR kod. Model 1 je orginalni dizajn, a Model 2 je nastao na temelju Modela 1 kako bi se povećao kapacitet i poboljšalo čitanje QR kodova sa zakrivljenih površina. Micro QR kod je dizajniran kako

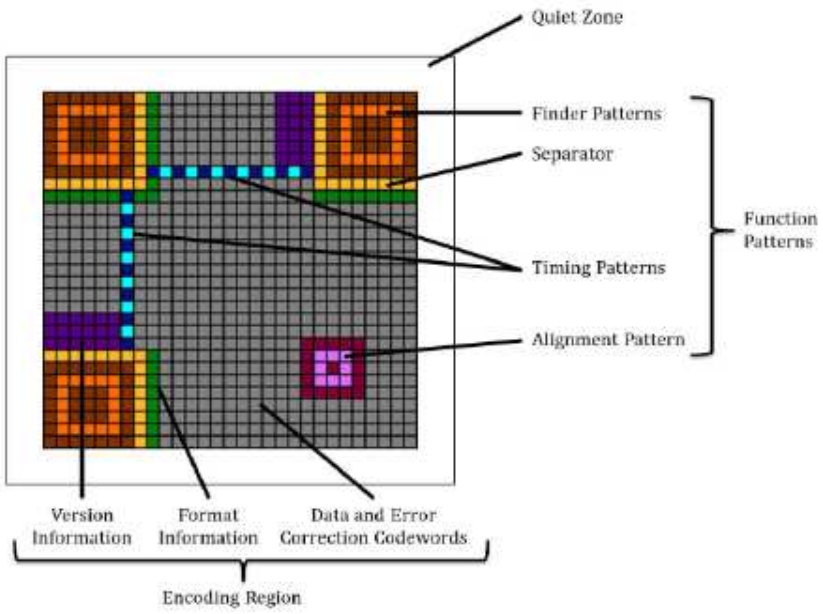
bi zauzimao što manju površinu. Zbog toga Micro QR kod ima samo jedan uzorak traženja (engl. *finder pattern*) za razliku od Model 2 QR koda koji ih ima tri. Logo QR kod sadrži logotip ili simbol unutar matrice. iQR kod je verzija QR koda koja ne mora nužno biti kvadrat, već može biti i pravokutnik. iQR kod je puno efikasniji u enkodiranju podataka od Modela 2, ali je njegova uporaba još uvijek ograničena i većina ga pametnih telefona ne podržava (Taylor, 2017). EQR kod je kriptirani QR kod tj. to je QR kod koji sadrži kriptirane podatke.



Slika 3.2. Verzije QR koda (Tiwari, 2016)

3.1. Struktura QR koda

QR kod je matrica koja se sastoji od određenog broja ćelija SxS. Svaka od tih ćelija može biti popunjena ili prazna. Određene ćelije su uvijek popunjene na isti način i te dijelove nazivamo funkcionalnim uzorcima. Funkcionalni uzorci služe kako bi detektori QR kodova mogli ispravno identificirati i orijentirati kod za dekodiranje (Tiwari, 2016). Postoje četiri tipa funkcionalnih uzoraka: uzorak traženja (engl. *finder pattern*), separator, uzorak vremena (engl. *timing pattern*) i uzorak poravnavanja (engl. *alignment pattern*). Ostale ćelije služe za enkodiranje verzije, formata informacije i same informacije. Dodatno, matricu ćelija okružuje četiri (ili više) ćelija široka prazna površina koja se naziva tiha zona (engl. *quiet zone*). Tiha zona služi kako bi odvojila QR kod od okoline i spriječila da tekst ili simboli koji okružuju QR kod utječu na dekodiranu informaciju. Slika 3.3. prikazuje strukturu QR koda.



Slika 3.3. Struktura QR koda (Tiwari, 2016)

Uzorak traženja služi kako bi skener QR kodova mogao uspješno orijentirati QR kod. Uzorci traženja uvijek isto izgledaju i uvijek se nalaze na tri pozicije u QR kodu: gornjem desnom kutu, gornjem lijevom kutu i donjem lijevom kutu. Dizajnirani su s idejom da je pojavljivanje skupa ćelija identičnih uzorku traženja u ostalim dijelovima QR koda vrlo malo vjerojatno (Tiwari, 2016).

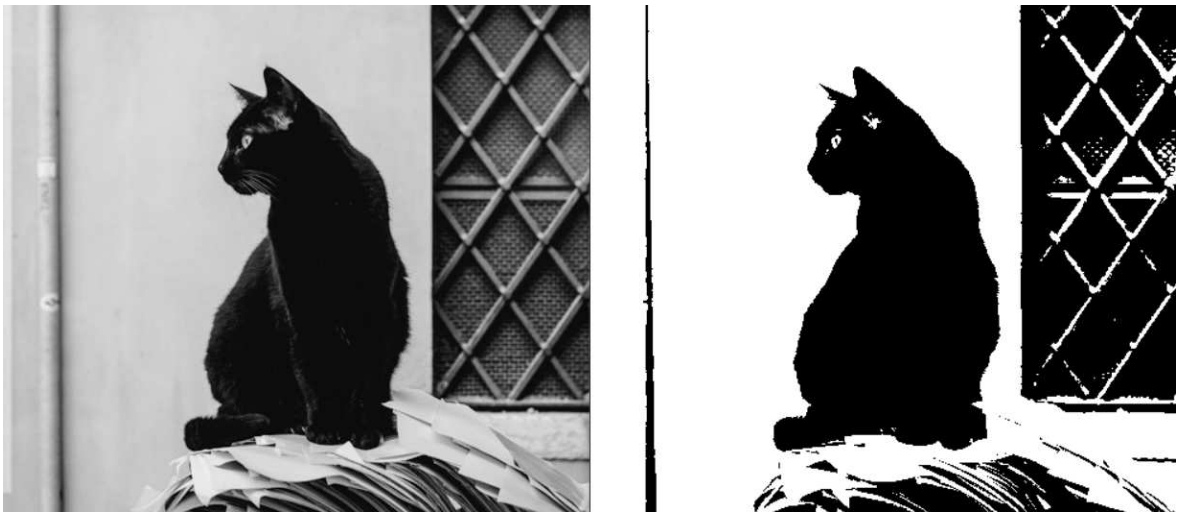
Separatori su prazne linije ćelija koje služe za odvajanje uzoraka traženja i ostatka QR koda (Tiwari, 2016).

Vremenski uzorci služe kao pomoć dekoderu u određivanju centralne koordinate QR koda (Soon, 2008). To je posebno važno u slučajevima kada je QR kod izobličen npr. ako se nalazi na zakrivljenoj površini. Svaki QR kod ima dva vremenska uzorka: jedan vertikalni i jedan horizontalni. Oni se sastoje od niza ćelija koje alterniraju između prazne i pune ćelije.

Uzorak poravnavanja služi za ispravno poravnavanje QR koda i ispravljanje mogućih izobličenosti (Soon, 2008). Kao i uzorak traženja, uzorak poravnavanja je dizajniran sa idejom da se isti uzorak ćelije neće ponoviti u drugim dijelovima QR koda. Broj uzoraka poravnavanja ovisi o veličini QR koda (Tiwari, 2016).

4. Binarizacija slike

Binarizacija (engl. *binarization* ili *thresholding*) slike je postupak kojim se multitonalna crno-bijela slika pretvara u dvotonalnu crno-bijelu sliku. U multitonalnoj crno-bijeloj slici postoje mnoge nijanse sive dok u dvotonalnoj crno-bijeli slici postoje samo dvije nijanse: crna i bijela.



Slika 4.1. Primjer binarizacije slike

Najjednostavniji postupak binarizacije slike je uspoređivanje svakog piksela multitonalne crno-bijele slike s unaprijed određenom graničnom vrijednošću (engl. *threshold value*). Ako je vrijednost piksela manja od granične vrijednosti tada se vrijednost piksela postavlja na nulu, a ako je veća tada se postavlja na maksimalnu vrijednost (Piyadasa, 2022). Ovaj postupak možemo opisati sljedećom formulom:

$$P(x,y) = \begin{cases} 0 & P(x,y) < threshold \\ max & P(x,y) \geq threshold \end{cases} \tag{4.1}$$

gdje je $P(x,y)$ vrijednost piksela na (x,y) koordinati slike.

Razlikujemo dvije vrste tehnika binarizacija: globalne i lokalne binarizacije (Piyadasa, 2022). Kod globalnih binarizacija ista granična vrijednost koristi se za čitavu sliku, a kod lokalnih se slika prvo podijeli u manje dijelove te se granična vrijednost određuje za svaku pojedinu regiju.

Adaptivna Gaussova binarizacije je tip lokalne binarizacije gdje se granična vrijednost svakog piksela računa na temelju Gaussove sume susjednih piksela. Graničnu vrijednost možemo opisati sljedećom formulom:

$$threshold = G(n) - c \tag{4.2}$$

gdje je $G(n)$ Gaussova suma svih najviše n udaljenih susjednih piksela, a c je unaprijed zadana proizvoljna konstanta.

Binarizacije slike ima široku primjenu u različitim područjima. U skeniranju, analizi i digitalizaciji dokumenata koristi se kako bi se tekst odvojio od pozadine. Koristi se u alatima za prepoznavanje uzoraka gdje je važnija struktura objekta nego njegova boja i nijansa kao npr. prepoznavanje otiska prsta. Pomaže u dekodiranju informacija prilikom čitanja barkodova i QR kodova (deepai.org).

5. Implementacija

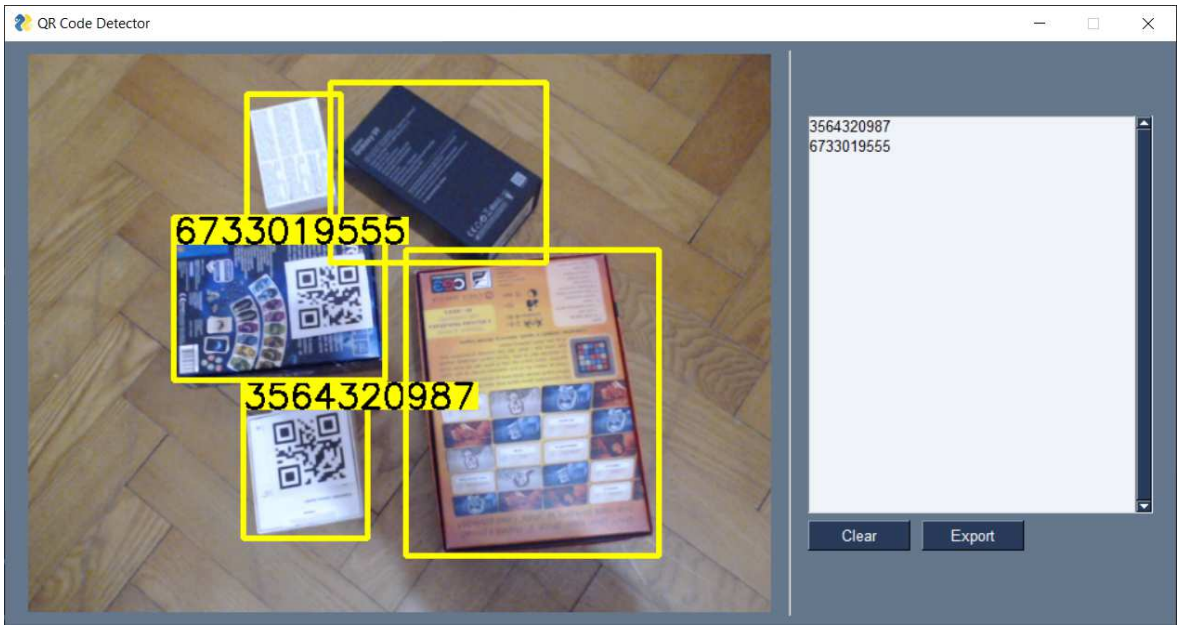
U ovom poglavlju predstavit će se program stvoren u sklopu ovog rada čija je svrha istovremena detekcija više objekata u stvarnom vremenu i dekodiranje QR kodova na njima. Objekte koje želimo detektirati su kutije s QR kodovima. Program na svakoj slici koju kamera snimi detektira sve kutije koje imaju QR kod te prikazuje njihove granične okvire i dekodirane vrijednosti QR kodova. Za implementaciju koda koristio se programski jezik *Python*. Kod je u cijelosti dostupan na sljedećem repozitoriju: `https://github.com/marin-rasic/boxqrcodedetector`.

Glavna petlja programa u svakom prolazu procesira sliku kamere: detektira objekte te detektira i dekodira QR kodove na njima. Slijedi pseudokod glavne petlje programa:

```
while True:
        frame = camera.read()
        detected_objects = detect_objects(frame)
        decoded_qr_codes = detect_and_decode_qr_codes(detected_objects)
        show_results(detected_objects, decoded_qr_codes)
```

U svakom prolazu petlje dohvaća se trenutna slika kamere. Zatim se provodi detekcija objekata što rezultira skupom graničnih okvira. Unutar svakog graničnog okvira traži se QR kod te se pokušava dekodirati. Uspješno dekodirani QR kodovi pridodaju se graničnom okviru kojem pripadaju. Na kraju se granični okviri s pripadajućim QR kodovima crtaju na ekranu. Ako QR kod nije detektiran ili uspješno dekodiran granični okvir će se i dalje nacrtati ali bez vrijednosti QR koda.

Za dohvaćanje slike s kamere i njihovu manipulaciju koristila se biblioteka OpenCV (`https://opencv.org/`). OpenCV (Open Source Computer Vision Library) je biblioteka otvorenog koda namijenjena za korištenje u području računalnog vida.



Slika 5.1. Primjer rada programa

5.1. Detekcija kutija

Model korišten za detekciju objekata je YOLOv8 model. YOLOv8 je najnovija verzija YOLO modela objavljena 10. siječnja 2023. godine. Razvila ga je i objavila američka softverska tvrtka Ultralytics. Model YOLOv8 je kod otvorenog tipa i slobodan je za komercijalnu upotrebu. U sklopu ovog projekta koristio se već trenirani modeli YOLOv8s. YOLOv8s je slobodno dostupan model trenirani na MS COCO skupu podataka opisanom u poglavlju 2.2.

Kako bi se model mogao iskoristiti za naše potrebe, YOLOv8s smo pretrenirali na našem skupu podataka tj. slikama koje sam sam fotografirao. U našem modelu postoji samo jedna klasa: kutija. Skup podataka sastoji se od 90 slika i 145 označenih objekata na tim slikama. Za upravljanje podatcima koristio se alat *Roboflow* (`https://roboflow.com/`). *Roboflow* je radni okvir za rad u području računalnog vida i korišten je u ovom radu za anotiranje slika tj. označavanja graničnih okvira objekata te pretprocesiranje. Pretprocesiranja se provelo kako bi se poboljšalo treniranje modela i povećao broj podataka za treniranje. Pojedine slike su duplicirane te su rotirane, pretvorene u crno-bijelu verziju ili je na njih dodan šum. Skup slika korištenih za treniranje našeg modela može se pronaći na sljedećoj poveznici: `https://universe.roboflow.com/projekt-xfn0j/box3-b55zo`.

Detekciju kutija provodi klasa *BoxObjectDetector*. Najvažnija funkcija te klase je funkcija *find_objects*(). Slijedi pojednostavljeni kod te funkcije:

```
def find_objects(self, image):
        if not is_right_image_size(image):
                resize_image(image)

        bounding_boxes = list()
        results = model(image)
        for box in results:
                bounding_boxes.append(box)

        return bounding_boxes
```

Funkcija će svakoj slici koja joj je predana promijeniti veličinu ako veličina ne odgovara unaprijed zadanoj. Veličina slike se može unaprijed zadati i za najbolje rezultate dimenzije slike bi trebala bi jednaka dimenzijama slika korištenih za treniranje modela.

Za pohranu informacija o graničnim okvirima koristimo klasu *BoundingBox*. Ona sadrži podatke o koordinatama graničnog okvira, vrijednosti pouzdanosti za taj granični okvir te vrijednosti QR koda koji je povezan uz taj granični okvir.

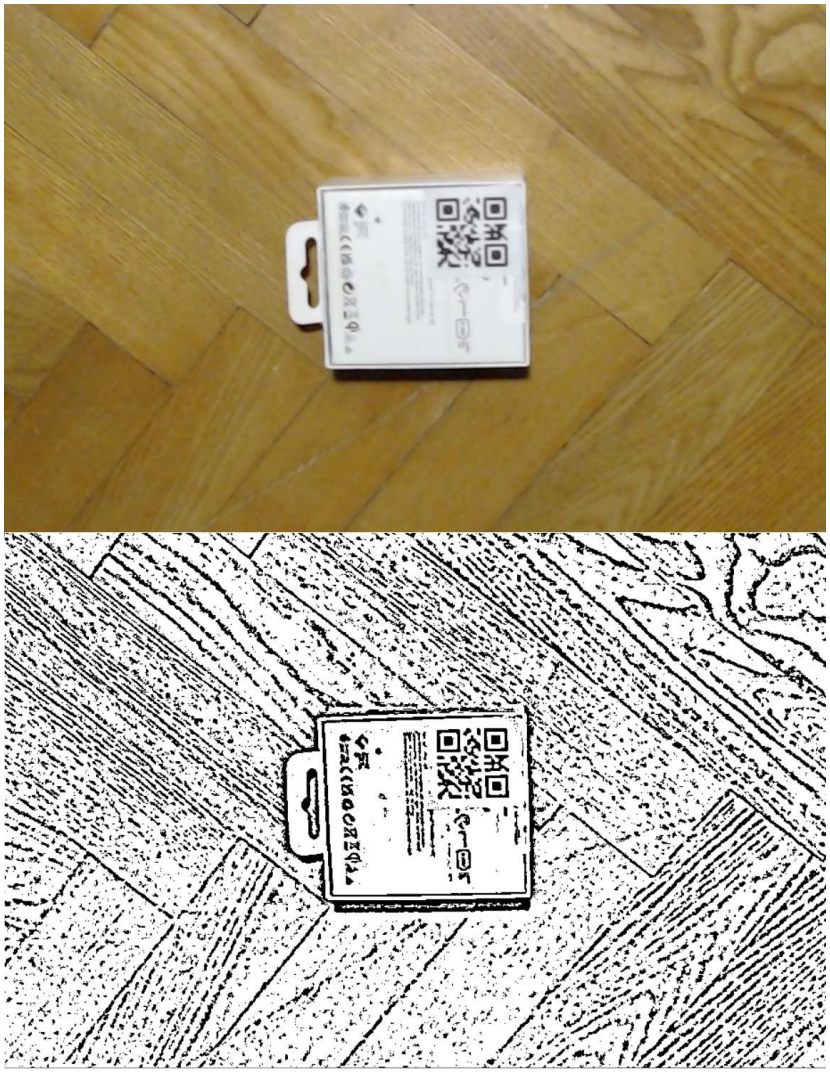
```
class BoundingBox:
        def __init__(self, x1: int, y1: int, x2: int, y2: int, conf: float):
                self._x1 = x1
                self._y1 = y1
                self._x2 = x2
                self._y2 = y2
                self._conf = conf
                self._qr_code = ""
```

5.2. Detekcija i dekodiranje QR koda

Za detekciju i dekodiranje QR koda koristimo biblioteku *pyzbar* (`https://github.com/NaturalHistoryMuseum/pyzbar/`). Kako bi osigurali da se QR kod uspješno detektira i dekodira provodimo pretprocesiranje slike. Koristimo činjenicu da su QR kodovi obično crno-bijeli te nad slikom provodimo binarizaciju. Koristimo adaptivnu Gaussova binarizaciju opisanu u poglavlju 4.

Za detekciju i dekodiranje QR kodova odgovorna je klasa *QRCodeFinder*. Njena glavna funkcija prima sliku i granični okvir te na temelju njih pokušava pronaći i dekodirati QR kod. Sljedi pojednostavljeni kod te funckije:

```
def detect_and_decode_qr_code_in_box(self, image, box):
        x1, y1, x2, y2 = box.get_coordinates()
        crop_image = image[y1:y2, x1:x2].copy()
        binariazed_image = self.image_binarization(crop_image)
        decoded_info = pyzbar.decode(binariazed_image)
        return decoded_info
```



Slika 5.2. Primjer binarizacije slike sa QR kodom

6. Rezultati

U ovom poglavlju prikazat ćemo rezultate implementacije našeg sustava detekcija kutija s QR kodovima. Glavni cilj našeg projekta je bilo ostvariti detekciju u stvarnom vremenu većeg broja objekata. Testirali smo brzinu i točnost programa za tri modela koje smo trenirali. Svaki model je treniran nad slikama različite rezolucije: 640x640, 480x480 i 320x320. Modeli testirani nad manjim rezolucijama su i sami manji. Tablica 6.1. prikazuje modele koje smo testirali. Testiranje je provedeno koristeći operacijski sustav *Windows 10* te procesor *Intel Core i5 12400*.

Tablica 6.1. Testirani modeli

Model	Rezolucija slike
A	640x640
B	480x480
C	320x320

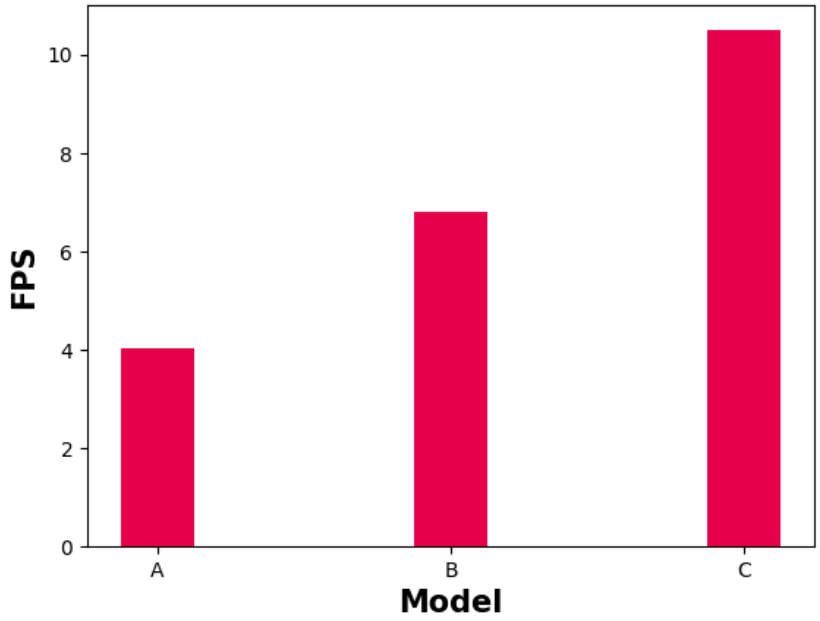
6.1. Brzina

Jedan od glavnih ciljeva projekta je bilo ostvariti detekciju objekata u stvarnom vremenu. Brzinu smo mjerili na temelju broja slika kamere koje program može obraditi unutar jedne sekunde ili FPS. Tablica 6.2. i graf na slici 6.1. prikazuje usporedbu brzina različitih modela. Što je rezolucija nad kojom je treniran model manja to je detekcija bila brža. Brzina rada programa minimalno je ovisila o broju objekata na slici. Nije primjećena nikakva značajna razlika u brzini kada se na slici nalazila samo jedna kutija ili njih 20.

Dodatno ubrzanje bi se moglo postići korištenjem boljeg procesora ili provođenjem detekcije na grafičkoj kartici te daljnjom optimizacijom koda i modela.

Tablica 6.2. Usporedba brzine

Model	Prosječni FPS
A	4.03
B	6.81
C	10.5



Slika 6.1. Usporedba brzina

6.2. Točnost

Točnost svakog modela smo testirali nad istim skupom podataka. Skup podataka se sastojao od devet slika sa sveukupno 44 kutija na njima. Svaka slika je uz kutije imala i par drugih predmeta poput knjiga, mobitela, čaša, voća i slično kako bi testirali hoće li naši modeli objekte koji nisu kutija prepoznati kao kutije. Tablica 6.3. prikazuje rezultate testiranja za svaku pojedinu sliku. Stupci "točno" i "netočno" govore koliko je točnih i netočnih graničnih okvira u rezultatima detekcije. Uvjet točnosti graničnog okvira definirali smo na sljedeći način:

$$T(bb) = \begin{cases} correct, & IoU(bb) \geq 0.5 \\ incorrect, & IoU(bb) < 0.5 \end{cases} \tag{6.1}$$

gdje *bb* predstavlja granični okvri, a IoU predstavlja presjek preko unije definiran u poglavlju 2.4.

Tablica 6.3. Rezultati testiranja točnosti

		Model A		Model B		Model C	
Slika	Broj kutija	Točno	Netočno	Točno	Netočno	Točno	Netočno
1	9	5	1	6	0	2	1
2	9	7	1	6	1	5	0
3	9	9	0	9	0	8	0
4	6	6	4	6	5	4	2
5	3	3	1	2	1	2	1
6	2	2	1	2	0	2	0
7	0	0	0	0	1	0	1
8	3	3	0	2	0	3	0
9	3	2	2	2	2	2	1

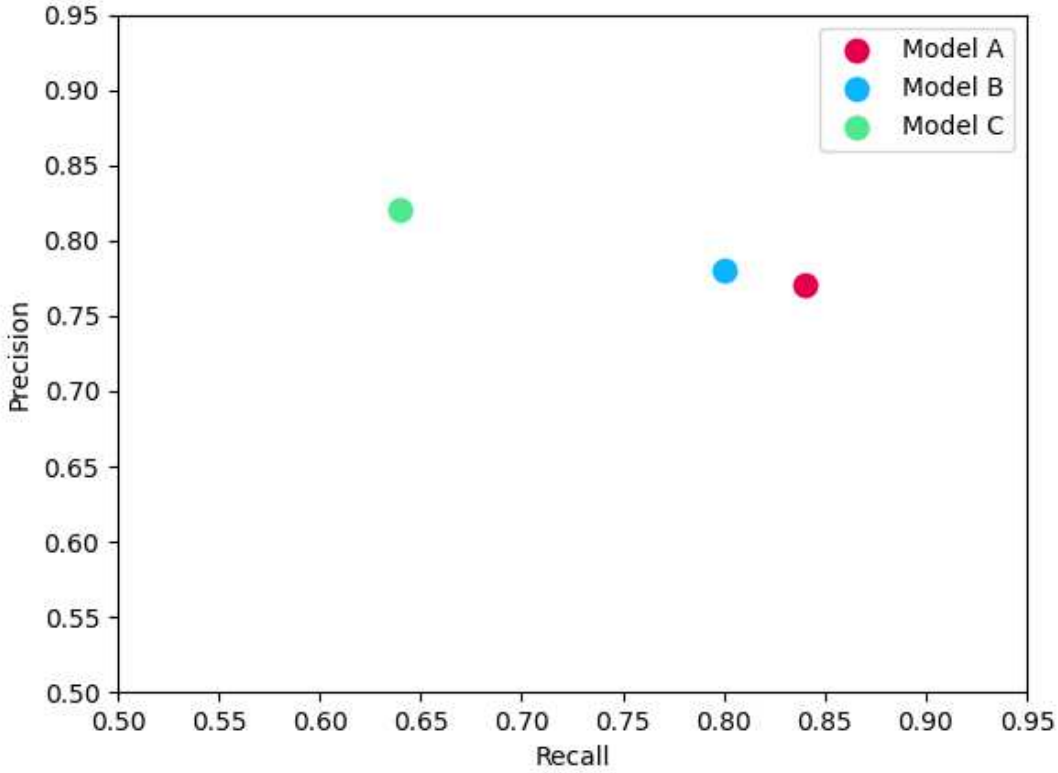
Najveći problem našim modelima su predstavljali objekti koji sliče na kutije ali to nisu npr. knjiga. Još jedan problem su predstavljali slučajevi gdje se dvije potpuno iste kutije nalaze jedna uz drugu. Tada bi modeli te dvije kutije prepoznali kao jednu.

Za svaki model smo izračunali i vrijednosti preciznosti i *recall*-a. Tablica 6.4. prikazuje vrijednosti preciznosti i *recall*-a za svaki pojedini model. Možemo vidjeti da modeli imaju slične preciznosti tj. omjer točnih detekcija i ukupnih detekcija im je podjednak. Velika razlika među modelima se vidi u *recall*-u. Model C ima znatno manju vrijednost *recall*-a od modela A i B. To znači da će model C objekte koji jesu kutija ignorirati u većem postotku nego modeli A i B.

Tablica 6.4. Rezultati testiranja točnosti

Model	Preciznost	Recall
A	0.77	0.84
B	0.78	0.80
C	0.82	0.64

Daljne povećanje točnost bi se prvenstveno moglo postići povećanjem broja slika korištenih za treniranje modela. Model bi se tada lakše nosio sa varijacijama među kutijama i bolje bi "naučio" ignorirati objekte koji sliče kutijama ali to nisu.



Slika 6.2. Graf točnosti i *recall*-a

7. Zaključak

Detekcija objekata jedan je od najznačajnijih problema područja računalnog vida. Razvoj detekcije objekata započeo je devedesetih godina 20. stoljeća. Današnji algoritmi za detekciju su bazirani na dubokom učenju i konvolucijskim neuronskim mrežama. Dvije najvažnije metrike detekcije objekata su brzina i točnost. Brzina je najvažnija u ostvarenju detekcije objekata u stvarnom vremenu. Algoritmi moraju biti efikasni i djelomično žrtvovati točnost kako bi postigli dovoljnu brzinu. YOLO model je jedan od modela za ostvarenje detekcije u stvarnom vremenu. To je jednofazni model što znači da sliku obrađuje samo jednom.

QR kodovi su široko raširen način kodiranja informacija. Kodirana informacija može biti broj, tekst ili URL. QR kodove možemo zamisliti kao matrice crno-bijelih ćelija. Svi QR kodovi imaju točno određen format kako bi ih detektori mogli detektirati i dekodirati.

U sklopu ovog rada stvoren je program za detektiranje kutija s QR kodovima. Glavni cilj je bilo postići detekciju u stvarnom vremenu te da se istovremeno može detektirati veći broj kutija. Za detekciju objekata koristio se YOLOv8 model. YOLOv8 model je najnovija verzija YOLO obitelji modela objavljena u siječnju 2023. godine. Implementirani program je uspješni pokazatelj da je detekcija u stvarnom vremenu većeg broja objekta moguća. Projekt bi se mogao dalje nadograditi proširivanjem skupa podataka korištenog za treniranje modela te optimizacijom dobivenog modela i programskog koda.

8. Literature

Amit, Y., Felzenszwalb, P., Girshick, R. Object Detection. Springer International Publishing, 2021.

Bengio, Y., Goodfellow, I., Courville, A. Deep Learning. MIT Press, 2016.

Binarization, 2019, https://deepai.org/machine-learning-glossary-and-terms/binarization, 16.1.2024.

Gad, A. F., Evaluating Object Detection Models Using Mean Average Precision (mAP), 2021, https://blog.paperspace.com/mean-average-precision/, 8.1.2024.

Hui, J., mAP (mean Average Precision) for Object Detection, 2018, https://jonathan-hui.medium.com/map-mean-average-precision-for-object-detection-45c121a31173, 8.1.2024.

Kumari, K., A Basic Introduction to Object Detection, 2023, https://www.analyticsvidhya.com/blog/2022/03/a-basic-introduction-to-object-detection/, 6.1.2024.

Max-pooling / pooling, 2018, https://computersciencewiki.org/index.php?title=Max-pooling_/_Pooling, 10.1.2024.

Mishra, M., Convolutional Neural Networks, Explained, 2020, https://towardsdatascience.com/convolutional-neural-networks-explained-9cc5188c4939, 10.1.2024.

Padilla, R., Passos, W. L., Dias, T. L. B., Netto, L., da Silva, E. A. B. A Comparative Analysis of Object Detection Metrics with a Companion Open-Source Toolkit. Electronics. 10, 3(2021)

Phung, V. H., Rhee, E. J. A High-Accuracy Model Average Ensemble of Convolutional Neural Networks for Classification of Cloud Image Patches on Small Datasets. Applied Sciences. 9, 21(2019)

Piyadasa, T. D., Image binarization in a nutshell, 2022, https://medium.com/@tharindad7/image-binarization-in-a-nutshell-b40b63c0228e, 15.1.2024.

Rastogi, V., Fully Connected Layer, 2023, https://medium.com/@vaibhav1403/fully-connected-layer-f13275337c7c, 11.1.2024.

Redmon, J., Divvala, S., Girshick, R., Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, SAD, (2016), str. 779-788.

Redmon, J., Farhadi, A. YOLOv3: An Incremental Improvement. CoRR. abs/1804.02767, (2018)

Rosebrock, A., Intersection over Union (IoU) for object detection, 2016, https://pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/, 21.1.2024.

Soon, T. J. QR code. Synthesis Journal. (2008), str. 59-78.

Taylor, L., What is An iQR Code?, 2017, https://qrcode.meetheed.com/question40.php, 25.1.2024.

Tiwari, S. An Introduction to QR Code Technology. 2016 International Conference on Information Technology (ICIT), Bhubaneswar, Indija, (2016), str. 39-44.

Yadav, N., Binay, U. Comparative Study of Object Detection Algorithms. International Research Journal of Engineering and Technology (IRJET). 4, 11(2017), str. 586-591

Zaidi, S. S. A., Ansari, M. S., Aslam, A., Kanwal, N., Asghar, M. N., Lee, B. A Survey of Modern Deep Learning based Object Detection Models. CoRR. abs/2104.11892, (2021)

Zou, Z., Shi, Z., Guo, Y., Ye, J. Object Detection in 20 Years: A Survey. CoRR. abs/1905.05055, (2019)

Sažetak

Sustav za detekciju i identifikaciju većeg broja proizvoda putem QR-kôda

Marin Rašić

Detekcija objekata je jedan od najznačajnijih problema područja računalnog vida. Moderni modeli detekcije objekata temelje se na dubokom učenju i neuronskim mrežama. Područje detekcije objekata se susreće s mnogim problemima, a jedan od njih je detekcija u stvarnom vremenu većeg broja objekata. QR kodovi su široko rasprostranjeni način kodiranja informacije. U sklopu ovog rada stvoren je program koji koristi YOLO model detekcije objekata za ostvarenje paralelne detekcije više objekata s QR kodovima u stvarnom vremenu.

Ključne riječi: detekcija objekata; konvolucijske neuronske mreže; YOLO model; QR kod

Abstract

System for detection and identification of multiple products via QR code

Marin Rašić

Object detection is one the most significant problems in the field of computer vision. Modern object detection models are based on deep learning and neural networks. The field of object detection faces many problems, one of which is real-time detection of a large number of objects. QR codes are a widespread way of encoding information. As part of this paper, a program was created that uses YOLO object detection model to achieve parallel detection of multiple objects with QR codes in real time.

Keywords: object detection; convolutional neural networks; YOLO model; QR code