ALGORITMI STROJNOG UČENJA NA PRIMJERU DETEKCIJE I OČITAVANJA AUTOMOBILSKIH TABLICA

Filip Matić

Prirodoslovno-matematički fakultet Split, Podatkovna znanost

E–mail: *fmatic@pmfst.hr*

Sažetak

Cilj ovog rada je istraživanje i kreiranje sustava za detekciju automobilskih tablica i očitavanje znakova s istih. Želimo kreirati sustav koristeći tehnologije zasnovane na umjetnoj inteligenciji i strojnom učenju, odnosno koristeći YOLO algoritam i TesseractOCR zasnovane na dubokim neuronskim mrežama i LSTM mrežama. Kreiran je eksperimentalni program koji demonstrira mogućnosti strojnog učenja na ovu temu, te su uspoređeni različiti modeli za različite korake u svrhu optimalnog iskustva. Modeli i metode su međusobno uspoređeni koristeći razne metrike za brzinu, preciznost i greške modela računalnog vida.

Keywords: computer vision, convolutional neural networks , license plate detection, long short term memory networks, optical character recognition

1. UVOD

Računalni vid je polje računarstva koje se fokusira na imitaciju kompleksnosti ljudskog vida kako bi omogućili računalima identifikaciju i procesiranje vizualnih signala, na sličan način ljudskom razumijevanju istih podražaja. Zahvaljujući značajnom napredku u područjima umjetne inteligencije, dubokog učenja i neuralnih mreža, računalni vid je eksponencijalno uznapredovao i u mogućnosti je čak i prestići ljudsko oko kod određenih zadataka detekcije i klasifikacije. [1]

Jedna od zanimljivijih primjena računalnog vida na svakodnevne probleme je automatska detekcija registracijskih tablica (kroz rad često skraćena kao ALPR ili ANPR). ALPR je postala bitna tema istraživanja još od 1990.-ih kada su se prvi put pojavili znanstveni radovi na tu temu. Od tada, proizvedeni su brojni različiti ALPR sistemi u svrhu naplate cestarine, granične kontrole, nadziranja prometnih zakona i prometa općenito. [2]

Klasični ALPR sistemi se sastoje od tri faze: detekcija tablica, segmentacija znakova i prepoznavanja segmentiranih znakova. Što se tiče prepoznavanja znakova, postoji novija praksa gdje se preskače proces segmentacije znakova, te se odmah prepoznaju znakovi na tablici, jer je segmentacija znakova sama po sebi zahtjevan zadatak čija uspješnost ovisi o sličnosti svih tablica koje se promatraju. Značajan utjecaj na segmentaciju pojedinih znakova imaju i svjetlosni uvjeti, šum i sjene, pa onda loša segmentacija potencijalno i pogorša rezultate samog prepoznavanja.

Unatoč brojnim otkrićima unutar polja računalnog vida, većina danas primijenjenih rješenja ovog problema i dalje su bazirani na metodama nevezanim za strojno i duboko učenje. Razlog tome je robusnost rješenja, odnosno njihova neprimjenjivost u "pravom životu" zbog potrebe za jakim procesorima i grafičkim karticama računala na kojem se proces vrti. [2]

Efikasan ALPR sistem može igrati bitnu ulogu u raznim primjenama u kontekstu inteligentnih rješenja unutar prometa. Uspješno prepoznavanje registracijskih tablica ima potencijal automatizirati značajan dio reguliranja prometa, bez obzira na neplanirane scenarije i neuobičajene oblike i boje tablica.

Uzevši u obzir koje zahtjeve ALPR sistem treba zadovoljavati, kroz ovaj rad napravljen je i testiran sistem baziran na YOLO (You Only Look Once) modelima za detekciju objekata u pravom vremenu.[3] Ultralytics je tim zadužen za razvoj YOLO modela, te je za vrijeme pisanja ovog rada najnoviji YOLOv8 model, dok je v9 u eksperimentalnoj fazi. YOLO je svojom prvom pojavom u 2016.toj godini revolucionirao područje računalnog vida svojim inovativ-

nim pristupom, gdje je modelu dovoljno samo jednom proći kroz mrežu fotografije (odakle i ime You Only Look Once), za razliku od dotadašnjih sistema koji su se morali izvršavati više stotina ili tisuća puta ovisno o fotografiji.[3]

YOLOv8 je YOLO verzija koja se koristi kroz ovaj rad, ujedno je i najnovija gotova verzija. YOLOv8 je pušten u korištenje u siječnju 2023. godine, te pruža pet različitih skalabilnih paketa ovisno o veličini podataka i kompleksnosti zadatka: YOLOv8n (nano), YOLOv8s (small), YOLOv8m (medium), YOLOv8l (large) and YOLOv8x (extra large). [4]

Za treniranje samog modela koristimo relativno velik skup fotografija, objedinjen iz više skupova podataka dostupnih na Roboflow internet stranici. Postignuti skup podataka uključuje fotografije tablica iz različitih kutova, pod raznim svjetlosnim uvjetima i s različitim lokacijama tablica na automobilu. Također skup podataka uključuje fotografije tablica s raznim verzijama tablica i pozadinskih boja, kao i fotografije tablica s motocikala i kamiona. Treniranje na ovaj način, s raznolikim skupom podataka, omogućuje detekciju tablica pri težim uvjetima.

Nakon same lokacije tablica, sistem odvaja područje interesa i poboljšava fotografiju koristeći OpenCV Python biblioteku. Nakon segmentacije područja interesa, samo prepoznavanje znakova se vrši koristeći TesseractOCR softver.[5] Tesseract je besplatan OCR (optical character recognitiont) softver otvorenog koda, te je 5.x.x. posljednja stabilna verzija u vrijeme pisanja ovog rada. Tesseract je od verzije 4.x.x. baziran na strojnom učenju, točnije na LSTM neuronskim mrežama. Za ALPR sistem je koristan zbog stalno rastućeg broja jezika i pisama koje podržava, kao i zbog mogućnosti čitanja nehorizontalnog teksta uspješno. Kako bi se provjerila uspješnost rada ALPR sistema, kroz članak je opisano i testiranje više modela i njihovi performansi.

2. PROBLEM DETEKCIJE REGISTARSKE TABLICE I PREPOZNAVANJA ZNAKOVA

Sustav za automatsku detekciju registarskih tablica na vozilima i očitavanja znakova s njih (ANPR) je rješenje brojnih prometnih pitanja u svrhu sigurnijeg i jednostavnijeg upravljanja prometom. Primjene su razne, uključujući automatizirani sustav za naplatu cestarine, upravljanje parkiralištima, brojanje prometa ali i praćenje ukradenih vozila. Postoje brojna rješenja za ANPR sustave, uključujući starije metode računalne grafike i obrade slike, do današnjih metoda strojnog učenja i računalnog vida. Prvi korak u procesu je lociranje tablice na vozilu i njena ekstrakcija, bilo u stvarnom vremenu ili na videozapisu. Nakon uspješne detekcije slijedi prepoznavanje znakova na tablicama. Najčešća rješenja uključuju implementaciju konvolucijskih neuronskih mreža (CNN) [6] i rekurentnih neuronskih mreža(RNN).[7] [8]

2.1. SRODNI RADOVI

U ovom poglavlju, kratko se spominjem srodnih postojećih radova na temu automatskog prepoznavanja automobilskih tablica. Također ukratko raspravljam radove na temu pojedinačnih dijelova procesa ANPR-a, te kako se znanje iz tih radova može koristiti kao korak u procesu mog projekta. Za istraživanje srodnih radova koristili su se sljedeći kriteriji:

- Radovi primjenjivi u stvarnom vremenu
- Radovi vezani za problem detekcije automobilskih tablica
- Radovi zasnovani na YOLO tehnologiji i/ili Tesseract tehnologiji
- Radovi koji objašnjavaju pozadinu YOLO i OCR modela
- Radovi napisani na engleskom jeziku

S obzirom da nije odabran neki ogledni znanstveni rad na kojem bi mogli bazirati ovaj, te do njegove godine objave zaključiti istraživanje, traženi su radovi od prve pojave YOLO modela (2015) do YOLOv9 (2024). Za pretraživanje baza podataka korišten je Google Scholar i Scemantic Scholar, s tim da Google Scholar nije isključivo namijenjen objavljenim znanstvenim radovima, pa se mora provjeravati pozadina rada. S obzirom na veliku količinu potencijalnih radova zbog velikog raspona godina, fokusirana je pretraga samo na radove čiji naslov i/ili kratak sadržaj ispunjavaju sve uvjete napisane iznad. Ovakvim sistemom je odlučeno u obzir uzeti radove navedene u tablici niže.

Naslov rada	Godina objavljivanja	Autor/i	Jezik rada	Teme rada	Kratak opis	Nedostatak
A Robust Real-Time Automatic License Plate Recognition Based on the YOLO Detector	2018.	Laroca et al.	Engleski	Računalni vid i prepoznavanje uzoraka	Automatska detekcija automobilskih tablica zasnovana na YOLOv2 modela i CNNu za prepoznavanje znakova	Od YOLOv2 verzije je postignut velik razvoj i napredak u svijetu računalnog vida
An Efficient and Layout-Independent Automatic License Plate Recognition System Based on the YOLO detector	2021.	Laroca et al.	Engleski	Računalni vid i prepoznavanje uzoraka	Automatska detekcija automobilskih tablica zasnovana na YOLOv3 modelu i CNNu za prepoznavanje znakova	Od YOLOv3 verzije je postignut velik razvoj i napredak u svijetu računalnog vida
Reading Car License Plates Using Deep Convolutional Neural Networks and LSTMs	2016.	Li, Shen	Engleski	Računalni vid i prepoznavanje uzoraka	Pristup ANPR problemu koristeći duboke neuronske mreže, točnije 37-klasnu konvolucijsku neronsku mrežu.	Zbog nekorištenja segmentacije, model lošije radi na tablicama drugačijeg oblika i pozicije.
Number Plate Detection Using YOLOV4 and Tesseract OCR	2022.	Poorani et al.	Engleski	Računalni vid i prepoznavanje znakova	Pristup ANPR problemu koristeći tada najmoderniji YOLOv4 model i TesseractOCR	Razvoj YOLO tehnologije od godine objave ovog rada
License Plate Recognition System Based on Improved YOLOv5 and GRU	2023.	Shi, Zhao	Engleski	Računalni vid i prepoznavanje znakova	Pristup ANPR problemu koristeći YOLOv5 i Gated recurrent units(GRU)	Kompliciran pristup prepoznavanju znakova bez OCR softvera
An Ultra-Fast Automatic License Plate Recognition Approach for Unconstrained Scenarios	2023.	Ke, Zeng, Guo	Engleski	Prepoznavanje uzoraka	Detekcija registracijskih tablica koristeći YOLOv3-tiny model	Nedostatak OCR-a i testiranje samo na fotografijama

Slika 1: Kratak pregled literature

Mnogi radovi se diraju teme automatskog prepoznavanja tablica, ili samo dijela tog procesa. U trenutku pisanja ovog rada, aktualne opcije iz svijeta strojnog učenja za ovaj problem bi bile YOLOv8 i TesseractOCR . YOLOv8 nema službenu dokumentaciju, te samim time postaje zanimljiviji za istražiti i testirati na ovu temu. Također, većina ovakvih radova i projekata dostupnih na webu nisu primjenjivi u stvarnom vremenu i stvarnim scenarijima, te je detekcija tablica ograničena samo na jednu zemlju čineći ih globalno neprimjenjivim.

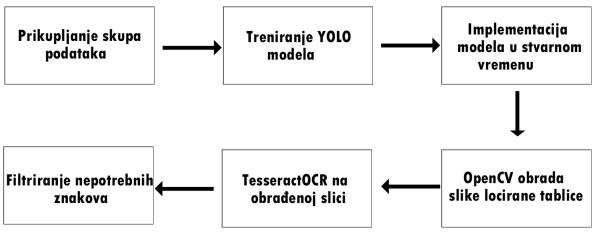
2.2. PREDLOŽENI SUSTAV

Proučavajući gore navedene srodne radove, stvorila se generalna ideja o rješavanju problema ANPR sustava koristeći metode strojnog učenja. Rješenju pristupamo kroz tri glavna koraka:

- Detekcija i lociranje tablice
- Obrada locirane tablice
- Prepoznavanje znakova s tablice

Treniranje modela za detekciju tablice vrši se koristeći oko 4500 slika skupljenih iz više javno dostupnih skupova podataka s Roboflow Universe web servisa. Korišten programski jezik je Python, uz biblioteku OpenCV. Za testiranje modela spajamo se na kameru u stvarnom vremenu, te se tablica locira i sprema u obliku fotografije koristeći funkcije OpenCV-a, kao i za obradu spremljene fotografije. Nakon obrade vrši se prepoznavanje znakova koristeći TesseractOCR i podaci se spremaju u .txt datoteku. Na slici 2 vidimo skiciran pregled sustava.

Ovakav pristup rješavanju problema ANPR sustava radi s metodama strojnog učenja u pozadini, te većinski koristi gotove funkcije i sustave. Modeli poput YOLO-a i Tesseracta znatno olakšavaju brojne probleme iz stvarnog svijeta vezane za računalni vid i smanjuju potrebu za dubokim razumijevanjem pojmova poput neuronskih mreža i LSTM-a. Unatoč tome, kroz rad je opisana pozadina ovih modela, kao i načini za njihovu optimizaciju i korištenje. Navedeni sustav bi trebao brzo i jednostavno prepoznavati automobilske tablice bez potrebe za internetskom vezom.



Slika 2: Vizualizirani predloženi sustav

3. KORIŠTENE TEHNOLOGIJE

Kroz ovo poglavlje objašnjene su tehnologije korištene za realizaciju ovog projekta. Kao što je prije spomenuto, za realizaciju projekta potrebna je detekcija, odnosno lociranje automobilske tablice, procesiranje prikupljene tablice i čitanje znakova. U svrhu realizacije ovih koraka koristili smo YOLOv8 model, OpenCV, TesseractOCR i korišten je programski jezik Python.

3.1. YoloV8

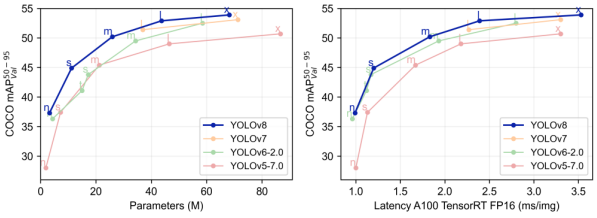
Yolo V8 je algoritam prepoznavanja objekata koji je pušten u korištenje u svibnju 2023. godine, te je prvenstveno stvoren kao nadogradnja na YoloV5. YOLOv8 je zasnovan na arhitekturi duboke konvolucijske neuralne mreže (CNN) koja je po mnogo stavki slična svojim prethodnicima. Razlike u odnosu na svoje prethodnike nalaze se u samoj arhitekturi modela, koja je detaljnije objašnjena kasnije u radu.

Osnovna arhitektura detektora objekata podijeljena je na tri dijela: okosnicu (backbone), vrat (neck) i glavu (head).

Okosnica je ključna u izdvajanju vrijednih značajki iz ulaznih slika, obično koristeći konvolucijsku neuronsku mrežu (CNN) obučenu na velikim skupovima podataka za klasifikaciju slika. Okosnica hvata hijerarhijske značajke različitih razmjera. Vrat je srednja komponenta koja povezuje osnovu s glavom.

YOLOv8 funkcionira tako što prvenstveno dijeli zaprimljenu sliku u mrežu ćelija. Za svaku ćeliju se predviđa set granica, zajedno s klasom vjerojatnosti za svaki granični kvadrat. Zatim se koristi non-maxima supresija (NMS) algoritam kako bi se filtrirale preklapajuće granice i izabrale granice s najvećom vjerojatnosti za svaki objekt na slici.[9] NMS algoritam je tehnika u detekciji objekata koja eliminira duple detekcije i odabire bitne prepoznate objekte.[10]

Za vrijeme pisanja ovog rada, postoji novija verzija YOLOv9, no i dalje je u razvojnom stadiju. YOLOv9 se u svom razvoju fokusira na nove značajke u arhitekturi modela, a to su programabilna gradijentna informacija (PGI) i generalizirana efikasna mreža za agregaciju slojeva (GELAN). PGI i GELAN imaju za cilj stvoriti poboljšanje protoka gradijenata, smanjiti gubitak informacija i održavati visoku točnost uz nižu računalnu složenost. [11] S obzirom na to da je v9 i dalje u razvojnom stadiju, u ovom radu odlučeno je koristiti v8 kao posljednju stabilnu verziju.



Slika 3: Usporedba YOLOv8 s ostalim YOLO modelima

Na slici 4 prikazana je detaljna arhitektura YOLOv8 modela. Vidljivi su prethodno spomenuti vrat, okosnica i glava. Okosnica je odgovorna za ekstrakciju značajki iz ulazne slike. Sastoji se od niza konvolucijskih slojeva i C2f blokova. Konvolucijski slojevi koriste filtere (jezgre) za ekstrakciju osnovnih značajki, dok C2f blokovi (Cross Stage Partial networks) omogućuju djelomično povezivanje značajki iz različitih slojeva, poboljšavajući učinkovitost mreže. Unutar okosnice postoje dvije faze, "split" i "concat", gdje se značajke prvenstveno dijele u "split" fazi, a zatim ponovno spajaju u "concat" fazi nakon prolaska kroz različite slojeve. [4] Vrat je dio modela koji dodatno kombinira i obogaćuje značajke primljene iz okosnice. Omogućuje detekciju objekata različitih veličina. Sastoji se ponovno od C2f blokova koji vrše istu funkciju, "upsample" slojeva koji povećavaju dimenzije značajki koristeći interpolaciju i omogućuju spajanje značajki različitih rezolucija i "concat" faze za spajanje značajki s različitih razina rezolucije. Glava je dio modela koji generira konačne predikcije, uključujući bounding boxove i klasifikacije objekata. Sastoji se od:

- Detect slojeva: Predviđa bounding boxove, klase i povjerenje za objekte u slici. Postavljeni su na različitim rezolucijama kako bi detektirali objekte različitih veličina.
- SPPF (Spatial Pyramid Pooling-Fast): Blok koji koristi višestruke konvolucijske jezgre za povećanje kontekstualnih informacija.

U nastavku su kratko objašnjene konvolucijske neuronske mreže u svrhu boljeg razumijevanja YOLO modela.

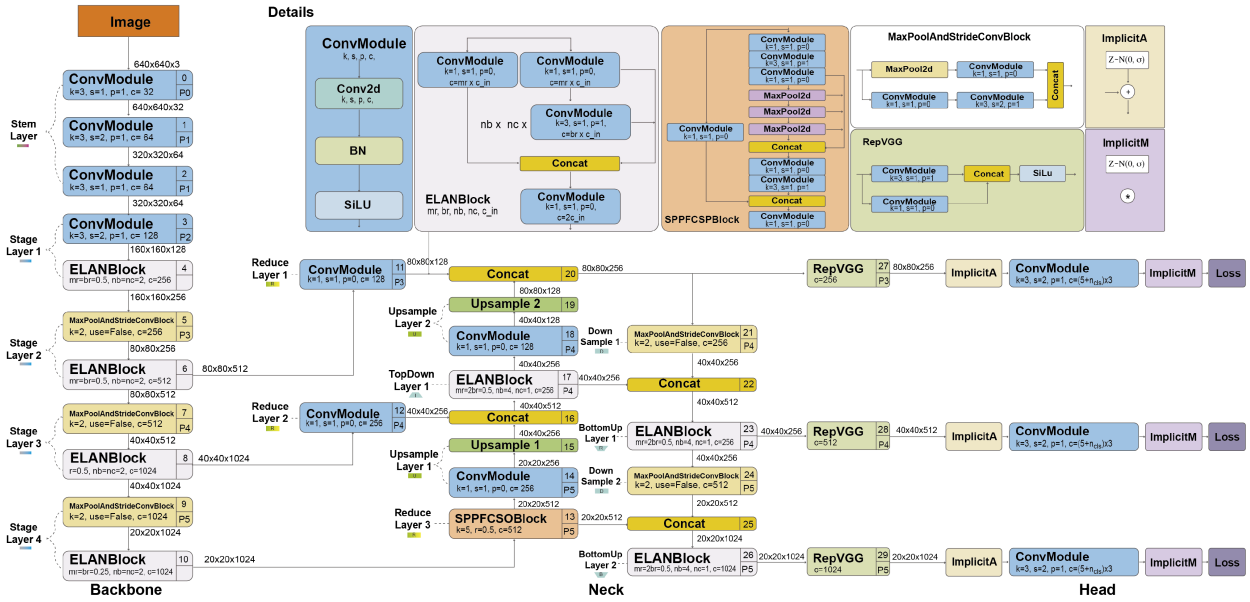
3.1.1. Konvolucijske neuronske mreže (CNN)

Duboko učenje (Deep learning) područje je strojnog učenja koje se bazira na treniranju složenih modela umjetne inteligencije, zvanih duboke neuronske mreže, čija je zadaća da kroz mnogo iteracija i ulaznih podataka nauče specifične karakteristike određenog skupa podataka.

Duboko učenje kao ulazne parametre koristi velike skupove podataka i složene algoritme optimizacije kako bi treniralo duboke neuronske mreže. Tijekom treninga, modeli se prilagođavaju podacima tako da automatski otkrivaju predodređene relevantne značajke i obrasce iz podataka, bez potrebe za ručnim određivanjem značajki. Glavna prednost dubokog učenja je sposobnost da iz velikih količina neobrađenih podataka nauči prepoznati ono što mu je zadano - bilo to prepoznavanje objekata, klasifikacija slika, prepoznavanje uzoraka, generiranje sadržaja i sl [13].

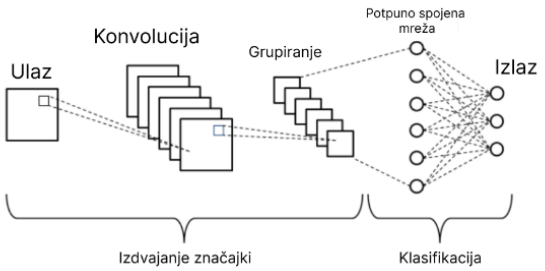
Tradicionalna metoda detekcije objekata se temeljila na ručno izrađenim karakteristikama, ali nije bila otporna na promjene osvjetljenja i nije pokazivala dobru sposobnost generalizacije.

CNN mreže su analogne tradicionalnim umjetnim neuronskim mrežama po tome što se sastoje od neurona koji se sami optimiziraju putem učenja. Uglavnom se sastoje od velikog broja međusobno povezanih



Slika 4: Detaljna arhitektura YOLOv8 modela
[12]

računalnih čvorova (neurona), koji rade isprepleteno u distribuiranom načinu kako bi kolektivno učili iz ulaza s ciljem optimizacije konačnog izlaza. [14]



Slika 5: Shematski prikaz arhitekture jednostavne CNN mreže [15]

Slika 5 prikazuje jednostavnu shematsku reprezentaciju nekog osnovnog CNN-a. Mreža se sastoji od 5 slojeva: ulaznog sloja, konvolucijskog sloja, grupacijskog sloja, sloja potpuno spojene mreže i izlaznog sloja.

Ulazni sloj određuje fiksnu veličinu za ulazne slike, čija se vrijednost može promijeniti ako je potrebno. U dijelu izdvajanja značajki kroz slojeve konvolucije i grupiranja, prolazi se više puta prije no što se dođe do dijela klasifikacije. U grupacijskom sloju se smanjuje veličina slike, uz zadržavanje sadržanih informacija. Nakon što se prođe konvolucija i grupiranje, izdvojene značajke se spajaju u potpuno spojenu mrežu, iz koje izlazi rezultat klasifikacije slike. [15]

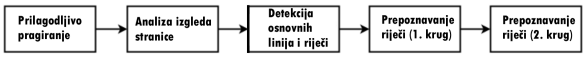
3.2. OpenCV & Python

S obzirom da je OpenCV biblioteka koja nema veze s aspektom strojnog učenja, ukratko ćemo objasniti njega i njegovu važnost u ovom projektu. Open Source Computer vision (OpenCV) je biblioteka/alat besplatan za akademsko i komercijalno korištenje. Radi se o biblioteci funkcija uglavnom namijenjenih za rad s računalnim vidom u stvarnom vremenu. Primjene ove biblioteke su široke, uključujući alate za 2D i 3D rad, sistem za prepoznavanje lica, gesti i razumijevanje kretnji, identifikaciju i segmentaciju objekata.[16] Kroz ovaj rad se koristi zbog brojnih korisnih predefiniranih funkcija za obradu slika u svrhu lakšeg segmentiranja slova s prepoznate tablice. Predefinirane funkcije olakšavaju pretvorbu slike iz RGB u Grayscale, dodatnu difuziju, pojačavanje kontrasta i slično. Projekt je pisan u Python jeziku. Radi se o jednom od najkorištenijih i najpopularnijih jezika za programiranje, sa širokim mogućnostima primjene.

3.3. Tesseract OCR

OCR (optical character recognition) je elektronska konverzija slika s tipkanim, pisanim ili printanim tekstom u strojno-enkodirani tekst. Tesseract je OCR softver koji se razvija od 1984. godine, te je 2005. HP pustio Tesseract u javnost kao softver otvorenog koda. Arhitektura Tesseracta je prikazana u 5 koraka na slici 6, te će više govora o pozadini i radu softvera biti u narednim poglavljima.[5]

Od Tesseract verzije 4 pa nadalje koriste se metode strojnog učenja, odnosno podsustav zasnovan na LSTM mrežama (Long Short-Term Memory). Tesse-



Slika 6: Arhitektura Tesseract OCR-a [5]

ractov sistem neuralnih mreža je integriran u softver kao "line recognizer".[5] Neuralne mreže pružaju bolje rezultate od starijih, osnovnih verzija Tesseracta, ali su i računalno zahtjevnije i teže za trenirati. U svrhu ovog projekta Tesseract nije treniran za nove fontove, zbog razlike u fontovima tablica od regije do regije. Testirane su posljednje dvije verzije koje implementiraju LSTM mreže.

4. REALIZACIJA SUSTAVA

Kroz ovo poglavlje ukratko je objašnjena realizacija predloženog sustava,usporedbe i rezultati.

4.1. Skup podataka

Za treniranje modela za prepoznavanje automobilskih tablica korišten je skup podataka od 5214 fotografija.[17] Skup podataka je skupljen od više javno dostupnih skupova na Roboflow web servisu. Skupljene fotografije su u JPG formatu, te su tablice već označene, nije bilo potrebe za dodatnim klasificiranjem. Skupljene fotografije objedinjuje tablice iz raznih regija, uključujući Europu, Ameriku,Aziju itd. U skupu podataka se nalaze i fotografije tablica drugih vozila osim automobila, kako bi model bio u stanju locirati i tablice na neuobičajenim lokacijama na drugim vozilima. Fotografije prikazuju tablice iz raznih kuteva, pod neoptimalnom svjetlosti i udaljenostima u svrhu optimizacije treniranog modela. Fotografije su podijeljene na skup za testiranje, validiranje i testiranje u količini 4893:540:266. Skup podataka se sastoji od fotografija veličine 640x640. Odabrani skup podataka je skupljen imajući na umu upute iz službenih uputa Ultralytics-a za treniranje YOLO modela. Prema istraživanju Ultralytics tima kvalitetan skup podataka mora sadržavati:

- više od 1500 fotografija po jednoj klasi
- raznolikost fotografija
- konzistentnost oznaka
- preciznost oznaka
- fotografije veličine 640x640 (u slučaju više manjih klasa po fotografiji model može profitirati i iz dimenzija 1280x1280) [18]

Slika 7: Mali dio skupa podataka

4.2. Treniranje modela

Nakon prikupljanja potrebnih podataka, potrebno je trenirati model za detekciju automobilskih tablica. Osim što YOLO nudi već izrađene modele, trenirane na COCO skupu podataka (Common Objects in Context), treniranje vlastitih modela je olakšano. Treniranje vlastitih modela povećava preciznost modela, eventualno ga ubrzava i pomaže pri izradi specifičnog programa. S obzirom da se skup podataka nalazi na Roboflow stranici, moguće je povezati skup podataka s Google Colab bilježnicom. Google Colab je besplatna Jupyter bilježnica, koja ne zahtjeva prethodno instaliranje i postavljanje, s dostupnim vanjskim resursima i oblakom za pohranu podataka. Roboflow nudi vlastitu python biblioteku u svrhu lakšeg treniranja modela, bez potrebe za preuzimanjem fotografija s njihovog oblaka. Na slici 8 vidimo način za povezivanje s vlastitim skupom podataka.

```
!pip install roboflow

from roboflow import Roboflow
rf = Roboflow(api_key=
project = rf.workspace("projekt").project("tablice-73he1")
version = project.version(5)
dataset = version.download("yolov8")
```

Slika 8: Povezivanje sa skupom podataka

YOLOv8 model je dostupan u više verzija, ovisno o broju parametara:

- YOLOv8n (nano)
- YOLOv8s (small)
- YOLOv8m (medium)
- YOLOv8l (large)
- YOLOv8x (extra-large)

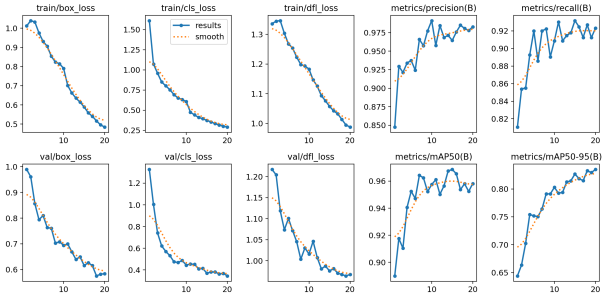
Na slici 9 prikazana je tablica službenih rezultata različitih modela iz YOLOv8 obitelji, njihovih performansi i osnovnih karakteristika. Modeli su korišteni za zadaće detekcije. Svi modeli su trenirani na bazama podataka fotografija dimenzija 640x640. Metrika za točnost, odnosno preciznost modela je Mean

Model	size (pixels)	mAP^{val} 50-95	Speed CPU ONNX (ms)	Speed A100 TensorRT (ms)	params (M)	FLOPs (B)
YOLOv8n	640	37.3	80.4	0.99	3.2	8.7
YOLOv8s	640	44.9	128.4	1.20	11.2	28.6
YOLOv8m	640	50.2	234.7	1.83	25.9	78.9
YOLOv8l	640	52.9	375.2	2.39	43.7	165.2
YOLOv8x	640	53.9	479.1	3.53	68.2	257.8

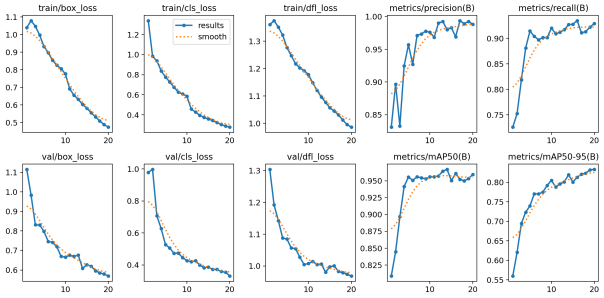
Slika 9: Službeni rezultati YOLOv8 modela [9]

Average Precision (mAP) na različitim pragovima od 0.50 do 0.95. Brzina procesa je izražena u milisekundama, koristeći navedeni CPU te zatim koristeći GPU NVIDIA A100. Parametri su izraženi u milijunima, te vidimo ogromnu razliku u parametrima kod nano i extra-large verzije. Posljednji stupac predstavlja "Floating Point Operations per Second", mjeren u milijardama, što predstavlja metriku računalne kompleksnosti modela.

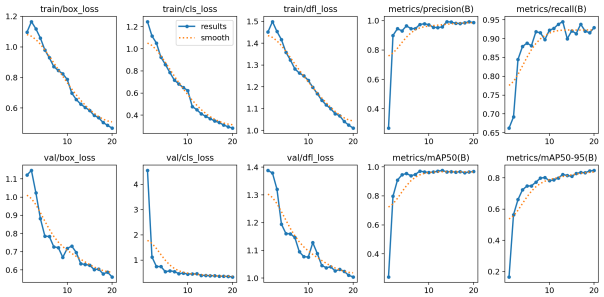
Za potrebe ANPR projekta testirane su 3 verzije modela: nano, small i medium. Odabrana su prva tri zbog nedostatka računalne snage za najveće verzije. Fotografije su veličine 640x640, a proces treniranja se vrtio na 20 epoha.



Slika 10: Metrike performansa nano modela



Slika 11: Metrike performansa small modela



Slika 12: Metrike performansa medium modela

Metrika	Objašnjenje
box_loss	Mjeri grešku kod predviđanja koordinata bounding boxa objekta.
cls_loss	Mjeri grešku kod predviđanja klase objekta
dfl_loss	Metrika koja mjeri balans klasa kod treniranja modela
metrics/precision	Odnos točno identificiranih objekata i ukupnih identifikacija
metrics/recall	Odnos točno predviđenih objekata i ukupnog broja predviđanja
mAP50	Mjera točnosti modela gledajući samo "lake" detekcije
map50-95	Mjera točnosti modela gledajući različite razine težine detekcije

Tablica 1: Objašnjenje metrika performansa YOLO modela

Iz gore vizualiziranih metrika, vidimo razlike u performansu između istog modela s različitim brojem parametara. Vidljivo je da s povećanjem broja para-

metara stabilnije se postižu bolji performansi, pogotovo kod mAP grafova. Gledajući preciznost i recall, manji modeli također postižu visoku točnost ali u manje slučajeva, dok je kod medium modela visoka preciznost stabilnija. Razlike u metrikama box_loss, cls_loss i dfl_loss su minimalne, zahvaljujući preciznom labeliranju skupa za treniranje.

IME MODELA	min_fps	max_fps
YOLOv8n	10	18
YOLOv8s	4	7
YOLOv8m	1	6

Tablica 2: Prikaz FPS-a za različite YOLOv8 modele

Nakon treniranja modela na 3 različite verzije, te usporedbe istih, zaključak je da je u ovoj primjeni nano model sasvim zadovoljavajuć. Prilikom testiranja većih modela u stvarnom vremenu, vidimo da je razlika u preciznosti nedovoljno velika da bi se tolerirao toliki nedostak sličica po sekundi.

YOLOv8 radi na principu konvolucijskih neuronskih mreža. Slika koju je model zaprimio prolazi kroz niz konvolucijskih slojeva koji izvlače značajke na različitim razinama apstrakcije. Prvi slojevi otkrivaju osnovne značajke poput rubova i tekstura, dok kasniji slojevi otkrivaju složenije oblike i objekte. Nadalje kombinira značajke s različitih razina rezolucije, omogućujući detekciju objekata različitih veličina. Niske razine rezolucije služe za detekciju većih objekata, a visoke za detekciju manjih. Model zatim predviđa koordinate bounding boxova kao relativne koordinate u odnosu na ulaznu sliku. Predikcija uključuje centar boxa (x,y), širinu(w) i visinu(h). Za svaki predviđeni bounding box, predviđa kojoj klasi objekt pripada, odnosno u našem slučaju pripada li klasi registracijskih tablica ili ne. Povjerenje (confidence score) koji YOLO nudi uključuje ukupnu sigurnost predikcije, kombinirajući sigurnost u postojanje objekta i sigurnost klasifikacije. [19]

4.3. Implementacija modela

Implementacija YOLO modela unutar Pythona je jednostavna. Nakon treniranja vlastitog modela unutar Google Colab bilježnice, na lokalno računalo preuzima se "best.pt" i uvozi u Python.

Iako model već označava bounding box i klasu, korisno je i locirati bounding box unutar programa kako bi bilo jednostavnije sačuvati prepoznatu tablicu. Za minimalan "confidence treshold" postavljen je 0.5, kako model ne bi bio zbunjen oblicima i bojama sličnim tablicama. Za trenutni eksperimentalni stadij projekta, ne postoji sistem automatizacije, te se slika tablice sprema manualno. YOLO prilikom paljenja kamere traži tablicu na svakom frame-u, a korisnik odabire trenutak za spremanje prepoznate tablice.

Slika 13: Testiranje personaliziranog modela na web-camu

4.4. PROCESIRANJE SLIKE

U svrhu najboljih performansi OCR softvera, potrebno je dodatno obraditi spremljene tablice. Obrađivanje se vrši koristeći spomenutu OpenCV biblioteku.

Napravljena funkcija prvenstveno konvertira sliku u grayscale, zatim je dvostruko povećava, primjenjuje Gaussian blur i binarizira koristeći Otsu metodu. Nakon toga koristi morfološke operacije za naglašavanje kontura i pokušava pronaći konture na slici. Konture se sortiraju, a za svaku konturu koja zadovoljava određene uvjete (odnos visine i širine, površina), izdvaja se regija interesa (ROI). Svaka regija interesa predstavlja jedan znak, a zbog mogućnosti više teksta na tablici, mora se zadovoljiti uvjet o visini kako bi bila kvalificirana kao znak s registracije. Konačno, prepoznati tekst se pohranjuje u varijablu plate_num, koja se vraća kao rezultat funkcije. U svrhu testiranja Tesseractovih mogućnosti, na kraju rada su uspoređeni rezultati bez segmentacije znakova i sa segmentacijom.

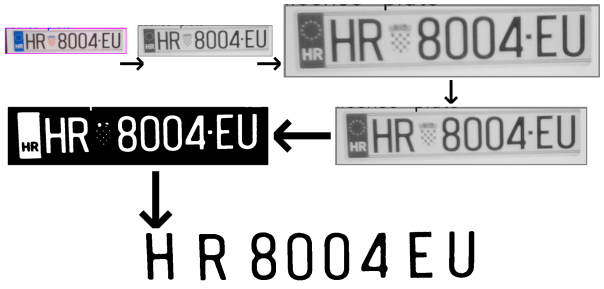
4.5. OČITAVANJE ZNAKOVA

Za očitavanje znakova koristi se prethodno spomenuti OCR softver Tesseract. Od verzije 4, Tesseract se zasniva na arhitekturi LSTM (Long Short-Term Memory) mreža [20], dok je najnovija verzija 5.0. Na 5 istih slika tablica testirao sam osnovni model verzije 4 i 5 u CDM-u.

Na baseline verziji Tesseracta vidimo bolji performans na novijoj verziji koja koristi LSTM. Vrijedi spomenuti da su registracijske tablice neuobičajen prikaz teksta, te je za bolje razumijevanje Tesseracta potrebno trenirati model na posebnim fontovima korištenim na tablicama. S obzirom da ovaj rad nije fokusiran na samo jednu regiju tablica, model nije treniran na novim fontovima.

Tablica	Tesseract 4	Tesseract 5
A12-A-345	A12-A-256	A12-A-5409
TA-003175	LA-GG3L75	ET A-003175
B147093	7B 1470930	B 147093
215-BG2	- -	215 BG2
MMA-7608	MMA-75G8	MMA-7608

Tablica 3: Rezultati Tesseract modela



Slika 14: Koraci obrade slike (sa segmentacijom)

Nakon obrađivanja spremljene tablice u koraku prije, obrađeni podaci se proslijeđuju Tesseractu.

Prilikom pozivanja Tesseractove funkcije textToImage, postoji mogućnost definiranja više parametara:

- Lista znakova koje tražimo
- Način segmentacije slike (–psm)
- Verzija OCR engine-a (–oem)

Za način segmentacije slike odabrana je –psm 10 vrijednost, što znači da tretira svaki dio slike koji smo segmentirali kao jedno slovo, a –oem 3 predstavlja korištenje zadanog načina rada gdje se koristi engine koji je dostupan (u ovom slučaju LSTM arhitektura).

Nakon detaljnog istraživanja, točna arhitektura i način rada Tesseracta s LSTM-om nije javno dostupna, ali se kroz više radova generalno može doći do nekih zaključaka o radu ovog OCR-a. Tesseract 4.0 i novije verzije koriste već spomenuti LSTM mrežnu arhitekturu kao zamjenu za proces od više koraka naveden ranije na slici 13. LSTM model korišten je zasnovan na CLSTM projektu (C++LSTM), koji je implementacija RNN mreže u C++ koristeći Eigen biblioteku. LSTM model u Tesseractu je opisan kao "text line recognizer", koji procesuira ulazne slike liniju po liniju.[21] Model ima složenu arhitekturu s više LSTM slojeva organiziranih u paralelnoj i/ili obrnutoj konfiguraciji. Posljednji sloj LSTM modela je SoftmaxLayer za klasifikaciju. Točan broj slojeva, njihove veličine i ostali arhitekturalni detalji nisu javno dostupni. [22] Bez točnih podataka nije moguće u potpunosti točno zaključiti pozadinu Tesseracta, ali od verzije 4 pa nadalje, zahvaljujući procesuiranju zasnovanom na LSTM-u i povećanim mogućnostima neuronskih mreža, OCR daje znatno bolje rezultate kod kompleksnijih tekstova, vertikalnog teksta i više jezika i fontova.

Prema službenoj dokumentaciji, OCR najbolje funkcionira u slučajevima gdje su slike u grayscale formatu, smanjenog šuma i na bijeloj pozadini s crnim tekstom. Na slici 14 su vizualizirani koraci obrade slike prije slanja Tesseractu, uključujući segmentaciju.

Na primjeru registracijske tablice iz Hrvatske prikazana je segmentirana slika pomoću YOLO modela, te njeno obrađivanje za proslijedu Tesseractu. U svrhu testiranja mogućnosti OCR-a, napravljena je i funkcija za segmentaciju znakova s tablice. Projekt je testiran sa –psm parametrom postavljenim na tretiranje slike kao jedne riječi, te sa segmentacijom znakova i parametrom postavljenim za prepoznavanje jednog znaka. U nastavku su prikazani rezultati testiranja.

Tablica	Sa segmentacijom	Bez segmentacije
HR 8004 EU	HR J EU	HR 8004 EU
BG 094 CM	BGM	BG 594 CM
WO 5869	4 WCR 6Q	W 5869
HV 360 08	HV 30 08	HV 360 0S
ZG 3591FZ	753FZ	MZG3591FZ
HB 041178	HBL1178	HB0411783

Tablica 4: Rezultati očitavanja znakova sa razvijenim softverom

Iz gornje tablice je vidljiva razlika prilikom prepoznavanja znakova, koristeći segmentaciju tablice i ne koristeći. U funkciji gdje se tablica segmentira, svaki segmentirani znak koristi Tesseract način rada za prepoznavanje samo jednog znaka, dok u drugom slučaju traži cijelu riječ.

Cijeli kod realiziranog projekta, uključujući datoteku sa segmentacijom tablice i bez segmentacije, Jupyter bilježnicu za treniranje YOLO modela i gotovi model, se nalazi na Github repozitoriju [23]

5. ZAKLJUČAK

Značajna otkrića u svijetu umjetne inteligencije i strojnog učenja direktno se koreliraju sa drugim podpodručjima poput računalnog vida, zbog čega vidimo veliki napredak u rastu i razvoju ovog dijela računarstva. Kroz detaljno istraživanje i realiziran program, kreirano je rješenje za detekciju automobilskih tablica i njihovo čitanje, zasnovano na algoritmima strojnog učenja. Primjena CNN-a i LSTM-a rezultira bržom detekcijom i točnijim prepoznavanjem, u svrhu automatizacije brojnih prometnih problema. Kreirani sustav se može primijeniti kao rješenje na ulazima u privatne parkinge, regulator

prometa na parkinzima i slično. Trenutno stanje programa je eksperimentalno, i kao takvo nema automatizirano spremanje tablica i ne funkcionira optimalno pri većim brzinama kretanja automobila, ali je moguće unapređenje kroz daljnje istraživanje.

6. LITERATURA

Literatura

[1] Towards Data Science. *Everything you ever wanted to know about computer vision*. Retrieved from https://towardsdatascience.com/everything-you-ever-wanted-to-know-about-computer-vision-heres-a-look-why-it-s-so-awesome-e8a58dfb641e. 2023.

[2] Rayson Laroca i dr. "A Robust Real-Time Automatic License Plate Recognition Based on the YOLO Detector". *2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE, srpanj 2018. DOI: 10.1109/ijcnn.2018.8489629. URL: http://dx.doi.org/10.1109/IJCNN.2018.8489629.

[3] Joseph Redmon i dr. "You Only Look Once: Unified, Real-Time Object Detection". *CoRR* abs/1506.02640 (2015.). arXiv: 1506.02640. URL: http://arxiv.org/abs/1506.02640.

[4] Juan Terven, Diana-Margarita Córdova-Esparza i Julio-Alejandro Romero-González. "A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS". *Machine Learning and Knowledge Extraction* 5.4 (2023.), str. 1680–1716. ISSN: 2504-4990. DOI: 10.3390/make5040083. URL: https://www.mdpi.com/2504-4990/5/4/83.

[5] R. Smith. "An Overview of the Tesseract OCR Engine". *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*. Sv. 2. 2007., str. 629–633. DOI: 10.1109/ICDAR.2007.4376991.

[6] Yann LeCun i dr. "Handwritten Digit Recognition with a Back-Propagation Network". *Advances in Neural Information Processing Systems*. Ur. D. Touretzky. Sv. 2. Morgan-Kaufmann, 1989. URL: https://proceedings.neurips.cc/paper_files/paper/1989/file/53c3bce66e43be4f209556518c2fcb54-Paper.pdf.

[7] J J Hopfield. "Neural networks and physical systems with emergent collective computational abilities." *Proceedings of the National Academy of Sciences* 79.8 (1982.), str. 2554–2558. DOI: 10.1073/pnas.79.8.2554. eprint: https://www.pnas.org/doi/pdf/10.1073/pnas.79.8.2554. URL: https://www.pnas.org/doi/abs/10.1073/pnas.79.8.2554.

[8] Hui Li i Chunhua Shen. "Reading Car License Plates Using Deep Convolutional Neural Networks and LSTMs". *CoRR* abs/1601.05610 (2016.). arXiv: 1601.05610. URL: http://arxiv.org/abs/1601.05610.

[9] Ultralytics. *YOLOv8 Architecture*. Retrieved from https://yolov8.org/yolov8-architecture/. 2023.

[10] A. Neubeck i L. Van Gool. "Efficient Non-Maximum Suppression". *18th International Conference on Pattern Recognition (ICPR'06)*. Sv. 3. 2006., str. 850–855. DOI: 10.1109/ICPR.2006.479.

[11] LearnOpenCV. *YOLOv9: Advancing the YOLO Legacy*. 2024. URL: https://learnopencv.com/yolov9-advancing-the-yolo-legacy/.

[12] Ultralytics. *Issue #189*. Retrieved from https://github.com/ultralytics/ultralytics/issues/189. 2023.

[13] Wang Zhiqiang i Liu Jun. "A review of object detection based on convolutional neural network". *2017 36th Chinese Control Conference (CCC)*. 2017., str. 11104–11109. DOI: 10.23919/ChiCC.2017.8029130.

[14] Keiron O'Shea i Ryan Nash. "An Introduction to Convolutional Neural Networks". *CoRR* abs/1511.08458 (2015.). arXiv: 1511.08458. URL: http://arxiv.org/abs/1511.08458.

[15] Phung i Rhee. "A High-Accuracy Model Average Ensemble of Convolutional Neural Networks for Classification of Cloud Image Patches on Small Datasets". *Applied Sciences* 9 (listopad 2019.), str. 4500. DOI: 10.3390/app9214500.

[16] Rahul R. Palekar i dr. "Real time license plate detection using openCV and tesseract". *2017 International Conference on Communication and Signal Processing (ICCSP)*. 2017., str. 2111–2115. DOI: 10.1109/ICCSP.2017.8286778.

[17] Filip Tablice. *ANPR Dataset*. Accessed: 2024-05-31. 2024. URL: https://app.roboflow.com/tablicefilip/anpr-d7qnq/deploy.

[18] Ultralytics. *Tips for Best Training Results*. https://docs.ultralytics.com/yolov5/tutorials/tips_for_best_training_results/. Accessed: 2024-05-14. 2024.

[19] Ultralytics. *Ultralytics/ultralytics: YOLOv8*. https://github.com/ultralytics/ultralytics. Accessed: 2024-05-14. 2024.

[20] Sepp Hochreiter i Jürgen Schmidhuber. "Long Short-Term Memory". *Neural Computation* 9 (1997.), str. 1735–1780. URL: https://api.semanticscholar.org/CorpusID:1915014.

[21] Docsumo. *Tesseract OCR: A Comprehensive Guide*. Accessed: 2024-05-29. 2022. URL: https://www.docsumo.com/blog/tesseract-ocr.

[22] Stack Overflow User. *What's the architecture of the LSTM engine in Tesseract 4 OCR?* Accessed: 2024-05-29. 2020. URL: https://stackoverflow.com/questions/63746804/whats-the-architecture-of-the-lstm-engine-in-tesseract-4-ocr.

[23] Filip Matic. *License Plate Recognizer*. https://github.com/maticfilip/License-Plate-Recognizer. Accessed: 2024-06-02. 2024.