SVEUČILIŠTE U ZAGREBU

**FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA**

ZAVRŠNI RAD br. 1481

**Sustav za detekciju vozila u video snimkama prometnica**

Antonio Glavaš

Zagreb, lipanj 2024.

SVEUČILIŠTE U ZAGREBU

**FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA**

ZAVRŠNI RAD br. 1481

**Sustav za detekciju vozila u video snimkama prometnica**

Antonio Glavaš

Zagreb, lipanj 2024.

Sadržaj

[1. Uvod 1](#_Toc168925888)

[2. Umjetna inteligencija 2](#_Toc168925889)

[2. 1. Strojno učenje 2](#_Toc168925890)

[2.2. Duboko učenje 3](#_Toc168925891)

[2.3. Računalni vid 3](#_Toc168925892)

[3. Umjetne neuronske mreže 5](#_Toc168925893)

[3.1. Neuroni 5](#_Toc168925894)

[3.2. Arhitektura neuronske mreže 6](#_Toc168925895)

[3.3. Učenje umjetne neuronske mreže 7](#_Toc168925896)

[3.4. Konvolucijske neuronske mreže 9](#_Toc168925897)

[4. Ogled korištenih tehnologija 12](#_Toc168925898)

[4.1. YOLO model za detekciju objekata 12](#_Toc168925899)

[4. 1. 1. YOLOv8 model 12](#_Toc168925900)

[4. 2. Programski jezik Python 13](#_Toc168925901)

[4.3. Google Colab 14](#_Toc168925902)

[4.4. Kaggle 14](#_Toc168925903)

[4.5 PyCharm 14](#_Toc168925904)

[4.6 Alati za razvoj programske potpore za web 14](#_Toc168925905)

[5. Implementacija, eksperimenti i rezultati 16](#_Toc168925906)

[5.1. Unaprijed trenirani YOLOv8 model 16](#_Toc168925907)

[5.2. YOLOv8 model treniran na odabranom skupu podataka 18](#_Toc168925908)

[5.2.1. Pregled rezultata treniranja modela 18](#_Toc168925909)

[5.3. Usporedba i evaluacija unaprijed treniranog i vlastitog modela 21](#_Toc168925910)

[Zaključak 24](#_Toc168925911)

[Literatura 25](#_Toc168925912)

[Sažetak 27](#_Toc168925913)

[Summary 28](#_Toc168925914)

# **1. Uvod**

# **2. Umjetna inteligencija**

Umjetna inteligencija je tehnologija koja omogućuje računalu rješavanje problema za koje je potrebna ljudska inteligencija kao što su razumijevanje govora, vid i analiza scene, prepoznavanje uzoraka, klasifikacija objekata u pripadne skupine itd. Umjetna inteligencija objedinjuje mnogo različitih znanstvenih disciplina poput računarske znanosti, statistike i analize podataka, inženjerstva programske potpore i sklopovlja, lingvistiku, neuroznanost, filozofiju i psihologiju. [1] Automatizacija raznih poslova, smanjenje faktora ljudske greške, brzina, točnost i efikasnost samo su neke od prednosti korištenja umjetne inteligencije. Područje kojim se bavi umjetna inteligencija je široko i obuhvaća grane poput strojnog učenja, robotike, računalnog vida, ekspertnih sustava, obradu prirodnog jezika itd.

## **2. 1. Strojno učenje**

Strojno učenje je grana umjetne inteligencije čiji je glavni cilj razvoj programa i algoritama na temelju skupa podataka bez eksplicitnog navođenja svih instrukcija koje bi program trebao izvršavati. Na raspolaganju se nalazi model koji je definiran parametrima potrebnim za zadaću koju izvršava, a učenjem modela upravo se ti parametri optimiziraju. Cilj učenja modela je razvijanje mogućnosti da predviđa svojstva na neviđenim, novim podatcima, dakle cilj strojnog učenja je izgraditi model koji dobro generalizira. [2]

Strojno učenje možemo podijeliti u tri vrste, a to su nadzirano, nenadzirano i podržano učenje. Nadzirano učenje definirano je treniranjem modela na označenom skupu podataka, dakle s definiranim ulazom i izlazom, a zatim se koristi za predikciju (klasifikacija u slučaju diskretnih vrijednosti ili regresija u slučaju kontinuiranih vrijednosti). Nenadzirano učenje koristi se kada imamo zadani skup podataka bez zadanih oznaka. Cilj takve metode je otkriti sličnosti i razlike, odnosno pravilnosti u podatcima i napraviti adekvatno grupiranje podataka. Posljednja metoda jest podržano učenje i ono se temelji na učenju kroz iskustvo; svaka dobra akcija je nagrađena, a loša akcija penalizirana.

## **2.2. Duboko učenje**

Duboko učenje je grana strojnog učenja koja se temelji na dubokim neuronskim mrežama. Duboke neuronske mreže su neuronske mreže s puno skrivenih slojeva, odnosno pokazalo se da je efikasnije proširiti mrežu na način da se koristi više slojeva s manje neurona nego manje slojeva s više neurona. Praktični primjeri korištenja dubokog učenja su autonomna vozila koja prepoznaju prometne znakove, okolinu, pješake i sl., obrambeni sustavi koji koriste duboko učenje za označavanje područja interesa na slikama satelita, medicinske slike za olakšanu ili automatsku dijagnostiku i sustavi za sigurnost koji detektiraju ako je čovjek ili neki drugi objekt u opasnome području. [3] Za provođenje dubokog učenja, potrebne su velike količine označenih podatka (npr. slike s vozilima i informacija o pripadnim koordinatama pojedinog vozila na predanoj slici) i visokokvalitetne grafičke procesorske jedinice, budući da je obrada i rad sa slikama računalno vrlo zahtjevan proces i zahtjeva mnogo resursa.

## **2.3. Računalni vid**

Računalni vid je grana računarske znanosti koja je fokusirana na omogućivanju računala da identificira i razumije objekte i ljude na slikama i video snimkama, odnosno cilj računalnog vida je replicirati na računalu način i razumijevanje onoga što ljudi vide. [4] Postoji mnogo načina na koji se koristi računalni vid od kojih su najčešći i najvažniji:

* Segmentacija slike – razdvajanje slike na particije interesantne za problem kojemu pristupamo.
* Detekcija objekata – identificiranje specifičnih objekata na slikama ili video snimkama kao što su npr. vozila i pri identifikaciji se objekt adekvatno uokviri.
* Raspoznavanje simbola – identifikacija brojeva, slova i ostalih simbola na slici te pretvorba u tekst kojeg razumiju računala.
* Prepoznavanje lica – napredna vrsta detekcije objekata koja se danas često koristi kao opcija pri biometrijskoj sigurnosti i cilj je ne samo prepoznati lice, nego i identificirati osobu koja se nalazi na slici.
* Detekcija rubova – tehnika koja se koristi za poboljšanje kvalitete slike ili identificiranja specifičnih objekata na slici.
* Detekcija uzoraka – proces prepoznavanja oblika, boja i ostalih vizualnih pojava koje se ponavljaju na slici.
* Klasifikacija objekata na slici – pri identifikaciji objekata na slici, svrstavaju se u pripadne klase, npr. pri detekciji vozila, treba odlučiti radi li se o automobilu, autobusu, kamionu i sl. [5]

# **3. Umjetne neuronske mreže**

Poznato nam je da se mozak sastoji od velikog broja bioloških neurona koji rade paralelno i glavna im je uloga primitak, obrada i prijenos informacije. Oponašanjem neuronske mreže kakva se nalazi u mozgu, smišljena je umjetna neuronska mreža koja omogućava rješavanje problema koji su presloženi da bi se riješili na klasični, algoritamski način. Umjetna neuronska mreža je skup međusobno povezanih elemenata koji imaju ulogu matematičkog modela čija se funkcionalnost temelji na biološkom neuronu i služe za prijenos te obradu podataka [6].

## **3.1. Neuroni**

Neuron je osnovna građevna jedinica umjetnih neuronskih mreža i svaki neuron, odnosno svaki čvor predstavlja matematički model. Prikaz strukture neurona nalazi se na slici Sl. 3.0.1. Ulazni parametri neuronske mreže pomnože se s pripadnim težinama i opcionalno zbroje s vektorom pristranosti, a zatim kumulativno sumiraju:

(3.1.)

Zatim se primjenjuje prijenosna funkcija čija je uloga odlučiti koliko je važna informacija koju nosi trenutni neuron za izračun daljnjih predikcija. Postoje tri glavne vrste prijenosnih funkcija u umjetnim neuronskim mrežama, a to su:

* Funkcija skoka – definira se prag koji odlučuje o aktivaciji neurona.

(3.2.)

* Linearne funkcije (funkcije identiteta) – funkcija ne radi nikakve transformacije, već samo prenosi ono što je dobiveno.

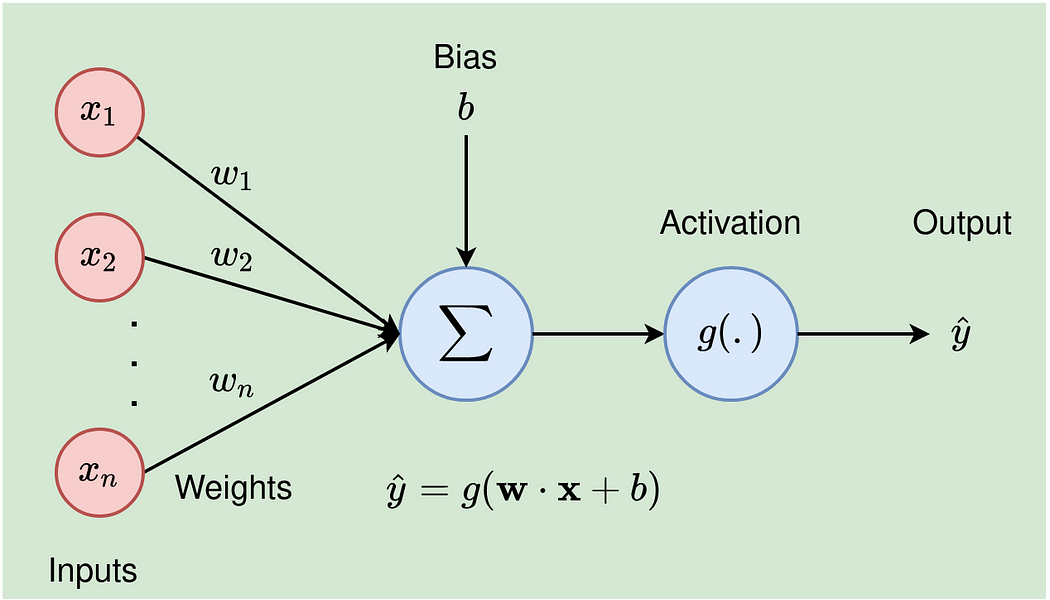
(3.3.)

* Nelinearne funkcije – dodavanjem nelinearnog sloja u neuronsku mrežu, omogućen je razvoj bolje i kompleksnije neuronske mreže s povećanom ekspresivnosti. Kao primjer nelinearnih prijenosnih funkcija često se primjenjuje logistička sigmoida:

(3.4)

Međutim, danas se sve više koristi ReLU (engl. Rectified linear unit) nelinearna funkcija koja omogućava treniranje dubljih mreža i računalno je učinkovitija za izračunati:

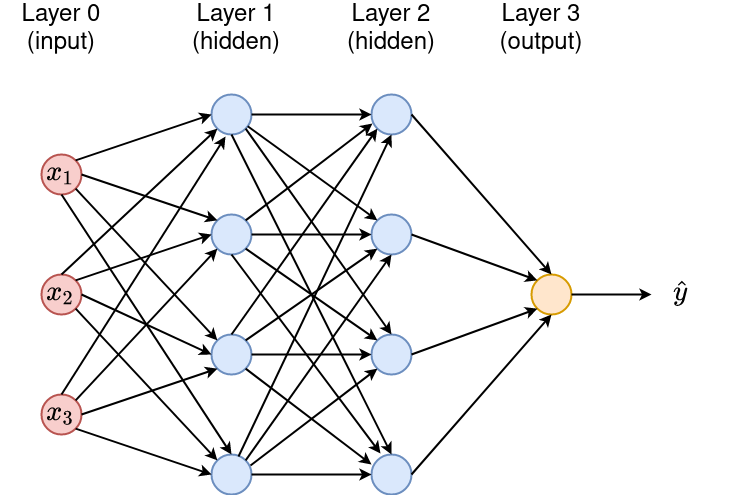
(3.5)



Sl. 3.0.1: Neuron u umjetnoj neuronskoj mreži [7]

## **3.2. Arhitektura neuronske mreže**

Neuronska mreža sastoji se od niza slojeva međusobno povezanih neurona kao što je to prikazano na slici Sl. 3.0.2. Prvi sloj je uvijek ulazni sloj i on ima onoliko čvorova kolika je veličina ulaznih parametara. U tom sloju ne izvršavaju se nikakve matematičke operacije, nego se samo prenose informacije u sljedeći sloj. Zatim slijedi skriveni sloj, koji ima jedan ili više slojeva, a taj broj ovisi o kompleksnosti problema koji se rješava. U skrivenom sloju izvršavaju se razne matematičke operacije koje su definirane u neuronima i prenose se iz sloja u sloj sve do posljednjeg, izlaznog sloja. Informacije na izlazu neuronske mreže predstavljaju ciljnu vrijednost, odnosno rezultat obrade zadanih podataka koji se može koristiti za donošenje odluke (u slučaju da se radi o klasifikaciji, dobili bi na izlazu o kojoj se klasi najvjerojatnije radi sudeći po zadanom skupu podataka) ili se prosljeđuje dalje, u iduću neuronsku mrežu.

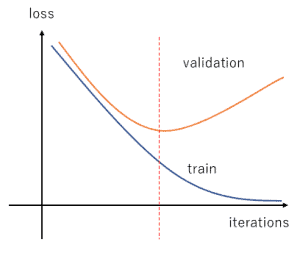


Sl. 3.0.2: Arhitektura neuronske mreže [7]

## **3.3. Učenje umjetne neuronske mreže**

Za učenje umjetne neuronske mreže, najprije treba pripremiti veliki skup podataka koji je točno označen kako bi dobili model koji može raditi kvalitetne predikcije budući da se ti podatci koriste se kao ulazni parametri u neuronskoj mreži. Prikupljeni skup podataka se najčešće dijeli u tri skupa podataka:

* Skup podataka za učenje – čini oko 70% priređenih podataka i koristi se za iterativno treniranje modela. Poželjno svojstvo skupa podataka za učenje je reprezentativnost budući da je cilj strojnog učenja napraviti model koji što bolje generalizira.
* Skup podataka za provjeru – čini oko 15% priređenih podataka i s njime se provjerava koliko je dobro generalizacijsko svojstvo mreže. Na slici Sl. 3.0.3 vidimo ovisnost pogreške o broju iteracija, odnosno epoha na skupu podataka za učenje i provjeru, dakle kako se povećava broj iteracija, tako se povećava i točnost na skupu podataka za učenje, model savršeno radi na takvom skupu. Međutim, u takvom slučaju javlja se problem kojeg nazivamo pretreniranost, što bi značilo da mreža gubi svojstvo generalizacije, stoga je cilj naći optimum (kojeg na grafu predstavlja crvena isprekidana linija) koji predstavlja kompromis između točnosti modela i svojstva generalizacije.



Sl. 3.0.3: Prikaz ovisnosti pogreške o broju epoha učenja [8]

* Skup podataka za testiranje – čini oko 15% priređenih podataka i koristi se za krajnju provjeru kvalitete mreže nakon što smo dovoljno istrenirali model i provjerili da radi dobro na skupu podataka za provjeru.

Učenje modela znači prilagođivanje težina i koeficijenta pristranosti pojedinih neurona u neuronskoj mreži kroz zadani broj epoha. Epoha predstavlja jedan obilazak cijelog skupa podataka za učenje i prilagodbu parametara neuronske mreže u ovisnosti o obiđenim podatcima kroz tu iteraciju. Kako bi neuronska mreža adekvatno prilagodila težine, potrebno je definirati funkciju pogreške, a jedna od takvih najčešće korištenih funkcija je srednje kvadratno odstupanje.

(3.6)

Funkcija izračunava prosjek kvadrata razlike između unaprijed poznate izlazne vrijednosti i izlazne vrijednosti koje izračuna neuronska mreža za zadani ulaz. Nakon definiranja funkcije pogreške, često se koristi postupak optimiranja težina neuronske mreže koji se temelji na izračunu gradijenta, odnosno iz parcijalnih derivacija moguće je saznati kako će se funkcija pogreške mijenjati ako mijenjamo ili smanjujemo težinu pojedinog neurona. Postupak učenja neuronskih mreža koji se temelji na prethodno opisanom postupku prilagodba težina naziva se postupak propagacije pogreške unatrag. [6] Neke od prednosti korištenja takvog postupka su činjenica da neuronska mreža ne treba imati nikakvo prethodno znanje za izvršavanje algoritma, nego se sve težine inicijalno postave na slučajne vrijednosti i podešavaju učenjem, efikasnost izvođenja algoritma i velika skalabilnost, odnosno postupak propagacije pogreške unatrag radi izvrsno na velikim količinama podataka.

## **3.4. Konvolucijske neuronske mreže**

Konvolucijske neuronske mreže su vrsta umjetnih neuronskih mreža koje se koriste za prepoznavanje uzoraka i objekata na slikama, dakle često se koriste u području računalnog vida. Kao i klasične umjetne neuronske mreže, konvolucijske neuronske mreže također imaju ulazni i izlazni sloj, međutim, kod konvolucijskih mreža skriveni sloj je nešto kompleksniji. U skrivenom sloju javljaju se konvolucijski sloj, aktivacijski sloj, sloj sažimanja (eng. pooling), sloj za izravnavanje i potpuno povezani sloj kao što vidimo na slici Sl. 3.0.4.

Slika na kojoj se prikazuje dijagram, skeč, crta, Plan

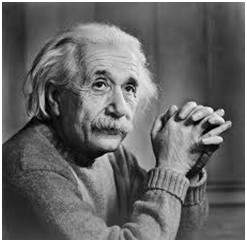
Opis je automatski generiran

Sl. 3.0.4: Arhitektura konvolucijske neuronske mreže [9]

Konvolucijski sloj je sloj koji iz zadanog skupa podataka izvršava ekstrakciju značajki kao što su rubovi i boje na slici uporabom raznih filtera od kojih su najpoznatiji Prewitt, Sobelov i Laplaceov operator. Kao primjer uzimamo Prewitt operator čija je vertikalna maska prikazana formulom (3.7) i horizontalna maska formulom (3.8), a na slici Sl. 3.0.5 nalazi se prikaz primjene horizontalnog Prewitt operatora na monokromatsku sliku. Ovisno o tome odaberemo li horizontalni ili vertikalni operator, Prewitt operator je izvrstan za izdvajanje rubova u smjeru koji je definiran maskom. Odabrani operator primjenjuje se na sliku na način da prolazi kroz određeni skup piksela (npr. 3x3) i izračunava se skalarni produkt.

(3.7)

(3.8)



Sl. 3.0.5: Primjena horizontalnog Prewitt operatora na monokromatsku sliku [10]

Konvolucijski sloj često dolazi u kombinaciji s aktivacijskim slojem koji ima ključnu ulogu u prijenosu i obradi podataka te uvodi element nelinearnosti u mrežu. Najčešće se kao prijenosna funkcija u aktivacijskom sloju uzima prethodno definirana (3.5) ReLU funkcija.

Zatim slijedi sloj sažimanja (eng. pooling) čija je glavna uloga smanjiti dimenzije dobivenih podataka kako bi se izračun u računalu ubrzao i olakšao te kako bi se izbjegao problem prenaučenosti. Postoje dvije vrste sažimanja koje se najčešće koriste a to su maksimalno sažimanje (eng. max pooling) i prosječno sažimanje (eng. average pooling). [9] Maksimalno sažimanje odabire najveću vrijednost na dijelu slike koji se trenutno obrađuje i odlično je za uklanjanje šuma. S druge strane, prosječno sažimanje odabire prosječnu vrijednost na dijelu slike koji se trenutno obrađuje i ima nešto niži performans u odnosu na maksimalno sažimanje.

Sloj za izravnavanje služi kako bi se podatci pretvorili u jednodimenzionalni vektor, što je pogodan oblik za daljnju obradu podataka u potpuno povezanom sloju.

Potpuno povezani sloj s primitkom prethodno obrađenih podataka izvršava klasifikaciju u slučaju da se radi o diskretnim izlaznim vrijednostima, odnosno regresiju ako se radi o kontinuiranim vrijednostima. [

# **4. Ogled korištenih tehnologija**

U ovome poglavlju nalazi se prikaz tehnologija korištenih za implementaciju sustava za detekciju vozila u video snimkama prometnica.

## **4.1. YOLO model za detekciju objekata**

YOLO (eng. You only look once) je jedan od najpopularnijih visokorazvijenih modela za detekciju objekata, a predstavio ga je Joseph Redmon 2016. godine. Glavna uloga YOLO modela je identifikacija i klasifikacija objekata jednim prolaskom kroz konvolucijsku neuronsku mrežu. Činjenica da YOLO model problem detekcije rješava kao regresiju, a ne klasifikaciju, čini ga mnogo bržim i efikasnijim od svojih prethodnika kao što je R-CNN (eng. Region-based Convolutional Neural Network). [12] R-CNN je također tip arhitekture koji se koristi u dubokom učenju za razne probleme poput detekcije objekata u području računalnog vida, a rade na način da se slika postepeno dijeli na regije koje bi mogle sadržavati objekte i pritom se ekstrahiraju relevantne značajke. [13] Osim brzine i efikasnosti, prednosti korištenja YOLO modela su visoka točnost u identifikaciji, bolja generalizacija u odnosu na prijašnje modele i činjenica da je model otvorenog izvora, što znači da ga svatko može slobodno koristiti i distribuirati.

U međuvremenu, YOLO modeli su znatno napredovali od 2016. kada su prvi puta predstavljeni, odnosno iz inicijalnog YOLO modela, razvili su se i YOLOv2, YOLOv3, pa sve do YOLOv10 koji je izašao u svibnju 2024. godine. Međutim, odabrani model koji će biti relevantan za izgradnju sustava za detekciju vozila je YOLOv8.

### **4. 1. 1. YOLOv8 model**

YOLOv8 model izašao je u siječnju 2023. godine kao inačica YOLO modela kojeg je razvila tvrtka Ultralytics, a koristi se za detekciju, klasifikaciju i praćenje objekata, segmentaciju slike. Prednosti korištenja YOLOv8 modela su činjenica da ima veću točnost predikcija pri identifikaciji objekata i napravljen je kao paket u programskom jeziku Python, što programerima olakšava korištenje. Osim toga, oko YOLOv8 modela razvila se velika zajednica, što znači da se može naći kvalitetna podrška pri korištenju tog modela. [14] Neke od inovacija koje čine YOLOv8 boljim odabirom od svojih prethodnika su uvođenje mehanizma za dodatno opažanje relevantnih dijelova slike, poboljšana točnost detekcije za male objekte i smanjena vremenska i prostorna kompleksnost programa uz zadržavanje točnosti.

Za treniranje i evaluaciju YOLOv8 modela koristi se COCO (eng. Common objects in context) skup podataka u kojemu se nalazi preko dvjesto tisuća slika iz realističnih situacija s više od osamdeset različitih klasa. COCO skup podataka je vrlo raznovrstan i često se koristi kao mjera kvalitete performansa modela u području računalnog vida.

## **4. 2. Programski jezik Python**

Sama implementacija sustava za detekciju vozila na video snimkama prometnica napravljena je u Python-u, koji je najpopularniji programski jezik za rad u području strojnog učenja, pa tako i računalnog vida. Python je objektno orijentirani programski jezik visoke razine koji se ne prevodi, nego za izvođenje koristi interpreter. Biblioteke u Python-u korištene za izvedbu sustava su:

* ultralytics – Biblioteka koja omogućava korištenje YOLOv8 modela, dakle moguće je koristiti unaprijed trenirani model, ali postoji i opcija treniranja modela od početka s vlastito odabranim skupom podataka. Osim treniranja, moguće je raditi i predikcije s ostvarenim modelom na slikama i video snimkama, a postoje još i neke zanimljive opcije poput brojanja objekta interesa ili procjena brzine vozila u prometu.
* cv2 – Biblioteka OpenCV je jedna od najpopularnijih biblioteka za područje računalnog vida, budući da nudi razne opcije za rad sa slikama poput prikaza slike, postavljanja praga, razne manipulacije poput promjene dimenzija, rotiranje, izrezivanje, filtriranje itd.
* flask - Radni okvir za razvoj programske potpore za web koji se nudi kao biblioteka u Pythonu. Koristi se za izradu dinamičnih, odnosno interaktivih web aplikacija što ga danas čini popularnim odabirom. Flask nudi programsku potporu za upravljanjem korisnikovih zahtjeva, prikazivanje HTML sadržaja i interakciju s bazom podataka te je je jednostavan za naučiti, koristiti i održavati. [22]

## **4.3. Google Colab**

Prije svega, treba napomenuti da je Jupyter bilježnica aplikacija koja nudi mogućnosti pisanja i pokretanja Python koda u blokovima koji se mogu izvršavati zasebno, a služi i za pisanje dokumentacije. Google Colab je internetski servis koji upravo nudi usluge Jupyter bilježnice bez potrebe za bilo kakvim instalacijama na računalo, što ga čini praktičnim za korištenje. Razlog koji čini Google Colab jednim od najpopularnijih servisa za korištenje u području strojnog učenja je besplatan pristup grafičkim procesorskim jedinicama (kao što je Nvidia Tesla T4, kojeg nudi besplatna verzija Google Colab-a), što je vrlo važno za efikasno treniranje na velikom skupu podataka. Osim toga, Google Colab ima već unaprijed instalirane popularne biblioteke za područje računalnog vida (kao što su TensorFlow ili PyTorch) i integriran je s Google Diskom što omogućava olakšani prijenos relevantnih datoteka.

## **4.4. Kaggle**

Kaggle je popularna internetska platforma s javno dostupnim skupovima podatka relevantnih za područja kao što su znanost o podatcima i strojno učenje. Osim toga, Kaggle organizira brojna natjecanja u kojima je cilj napraviti što bolji model za zadanu problematiku.

## **4.5 PyCharm**

PyCharm je jedno od najpopularnijih integriranih razvojnih okruženja (eng. IDE – Integrated development environment) za programski jezik Python. Razvila ga je tvrtka JetBrains i široko se koristi u područjima kao što su razvoj programske potpore za Web i znanost o podatcima.

## **4.6 Alati za razvoj programske potpore za web**

HTML, CSS, Javascript. HTML (eng. HyperText Markup Language) je označni jezik koji se koristi za izradu kostura web stranice i često dolazi u paru s CSS-om (eng. Cascading Style Sheets) CSS dodaje stil web stranici, odnosno koristi se kako bi vizualno uljepšali stranicu, dodali animacije, prilagodili izlged web stranice za različite uređaje itd. Ako želimo web stranicu učitini dinamičnom, u tom slučaju koristimo Javascript. To je skriptni jezik koji dodaje element interaktivnosti, a često se koriste i Javascript radni okviri na klijentskoj i poslužiteljskoj strani.

# **5. Implementacija, eksperimenti i rezultati**

Sustav za detekciju vozila na video snimkama prometnica može se implementirati korištenjem unaprijed treniranog YOLOv8 modela budući da je treniran na COCO skupu podataka u kojemu se nalaze klase relevantne za takav sustav kao što su automobil, motor, autobus i kamion. Osim toga, sustav je moguće ostvariti treniranjem YOLOv8 modela od početka na odabranom skupu podataka kroz određeni broj epoha. Glavni zadatci koje bi sustav trebao zadovoljavati uključuju identifikaciju i praćenje vozila na video snimci, prebrojavanje ulaznih i izlaznih vozila te procjena brzine vozila. U ovome poglavlju nalazi se prikaz i usporedba rezultata predikcija korištenjem i unaprijed treniranog modela, ali i modela treniranom na odabranom skupu podataka.

## **5.1. Unaprijed trenirani YOLOv8 model**

Kako bi imali referencu za usporedbu vlastito implementiranog modela za detekciju vozila, koristimo unaprijed trenirani YOLOv8 model koji se može vrlo jednostavno implementirati ako imamo instaliranu Python biblioteku *ultralytics* ili ako koristimo servis poput Google Colab-a. Nakon uvoza modela, možemo ga koristiti za predikcije na video snimaka i slikama te možemo zadavati dodatne parametre kao što su klase objekata koje nas zanimaju ili prag pouzdanosti ispod koje model ne bi detektirao objekte. Za ilustraciju prethodno opisane funkcionalnosti, pogledajmo isječak koda sa slike Sl. 5.0.1. Kao argumente za poziv funkcije za predikciju postavljamo respektivno put do željene datoteke nad kojom želimo da se izvrši detekcija, prag pouzdanosti, opcija želimo li spremiti rezultat, što u ovome slučaju želimo, i identifikacijski broj klasa koje nas interesiraju (automobil – 2, motor – 3, autobus – 5, kamion – 7). Kao rezultat dobivamo sliku Sl. 5.0.2 na kojoj vidimo da je detekcija automobila na slici uspješna i vozila su točno uokvirena.

|  |
| --- |
| Sl. 5.0.1: Demonstracija predikcije vozila korištenjem YOLO modela |

Slika na kojoj se prikazuje vanjski, oblak, zgrada, vozilo

Opis je automatski generiran

Sl. 5.0.2: Primjer detekcije automobila [20]

## **5.2. YOLOv8 model treniran na odabranom skupu podataka**

Kako bi omogućili YOLOv8 modelu prepoznavanje željenih objekata, najprije treba pronaći kvalitetan skup podataka koji sadrži reprezentativni uzorak objekata koje želimo detektirati. Međutim, za treniranje YOLOv8 modela nisu dovoljne samo slike, nego su potrebne i oznake u točno specificiranom formatu: (klasa objekta, x koordinata centra, y koordinata centra, širina, visina). Ako skup podataka nije označen, treba ga ručno anotirati korištenjem nekih od internetskih servisa poput CVAT [16] ili Roboflow [17].

Za treniranje modela u okviru ovoga rada koristimo unaprijed označeni skup podataka koji se može se pronaći na internetskoj stranici Kaggle: [18]. Skup podataka sastoji se od 1254 slike s pripadnim oznakama u obliku tekstualnih datoteka i podijeljen je u tri manja skupa podataka, odnosno u skupove za treniranje, validaciju i testiranje. Klase koje su podržane tim skupom podataka su vozilo hitne pomoći, autobus, automobil, motor i kamion.

### **5.2.1. Pregled rezultata treniranja modela**

Treniranje modela vrši se na platformi Google Colab korištenjem prethodno opisanog skupa podataka kroz 100 epoha. Kao rezultat treniranja dobijemo razne automatski generirane metrike za provjeru kvalitete modela i možemo postepeno pratiti razvoj modela kroz svaku epohu u procesu učenja.

Za početak, pogledajmo nekoliko metrika za napredak učenja po epohama. Na slikama Sl. 5.0.3 i Sl. 5.0.3 nalazi se nekoliko grafova koji predočavaju kako napreduje proces treniranja na skupu za učenje (slika Sl. 5.0.3) i na skupu za provjeru (slika Sl. 5.0.3). Budući da grafovi prikazuju ovisnost različitih gubitaka o epohama, s obzirom da je padajući graf, pretpostavljamo da učenje ide u dobrome smjeru. Na slikama su prikazane po dvije metrike: *box\_loss* koji predstavlja grešku prilikom crtanja okvira oko identificiranog objekta i *cls\_loss* koji predstavlja grešku prilikom klasifikacije objekata. Kako niti jedan graf nije krenuo stagnirati tijekom proteklih 100 epoha, pretpostavljamo da model još nije došao do stanja prenaučenosti, odnosno povećavanjem broja epoha, model bi postao još kvalitetniji.

Slika na kojoj se prikazuje tekst, crta, radnja, dijagram

Opis je automatski generiran Slika na kojoj se prikazuje tekst, dijagram, crta, radnja

Opis je automatski generiran

Sl. 5.0.3: Pogreška na skupu za učenje Sl. 5.4: Pogreška na skupu za provjeru

Kao iduće dvije metrike za evaluaciju modela uzimamo preciznost i odziv. Preciznost modela definirana je omjerom stvarnih pozitivnih identifikacija i sume stvarnih pozitivnih i lažnih pozitivnih identifikacija objekata. Odziv modela definiran je omjerom stvarnih pozitivnih identifikacija i sume stvarnih pozitivnih i lažnih negativnih identifikacija. Dakle, preciznost nam daje informaciju o postotku točnosti svih pozitivnih identifikacija, a odziv informaciju o postotku točnosti stvarnih pozitivnih identifikacija. [19] Kako bi model radio dobro, treba pronaći balans između preciznosti i odziva modela. Na slici Sl. 5. nalaze se dva grafa koja predstavljaju preciznost i odziv modela respektivno te možemo uočiti kako oba dvije metrike rastu s brojem epoha, što je dobar znak prilikom treniranja modela.

Slika na kojoj se prikazuje snimka zaslona, crta, radnja, dijagram

Opis je automatski generiran

Sl. 5.5: Prikaz preciznosti i odziva modela

Jedna od najpopularnijih metoda za evaluaciju­ modela u području strojnog učenja naziva se matrica zabune (eng. confusion matrix). Matrica zabune koristi se za vizualizaciju kvalitete predikcija pojedinih klasa, odnosno redovi u matrici zabune predstavljaju stvarnu klasu koja je predstavljena slikom, a stupci predstavljaju klasu koju je model identificirao. S obzirom na tu tvrdnju, pod pretpostavkom da se sve klase poredaju istim redoslijedom po redovima i stupcima, možemo uočiti kako bi idealna matrica zabune bila dijagonalna matrica, odnosno sve vrijednosti bi se nalazile na dijagonali. Matrica zabune za model koji prepoznaje vozila nalazi se na slici Sl. 5.. Možemo uočiti kako se pojedine klase poprilično dobro prepoznaju, ne uzimajući u obzir pozadinu. S obzirom da je puno primjeraka identificirano kao pozadina, a po prethodnim metrikama smo zaključili da se ne radi o prenaučenosti, možemo zaključiti kako bi model imao bolji performans kada bi se skup podataka za učenje proširio kvalitetno označenim slikama ili kada bi se povećao broj epoha učenja modela. Osim toga, najveći broj slika koje se nalaze u skupu podataka predstavljaju automobile, što znači da skup podataka nije najbolje balansiran, odnosno skup podataka bi trebalo proširiti sa slikama ostalih klasa.

Slika na kojoj se prikazuje snimka zaslona, Trokut, tekst, dijagram

Opis je automatski generiran

Sl. 5.6: Matrica zabune

## **5.3. Usporedba i evaluacija unaprijed treniranog i vlastitog modela**

Za početak, pogledajmo kako bi dva prethodno opisana modela identificirala objekte na istoj slici. Kao rezultat predikcije korištenjem unaprijed treniranog modela dobivamo sliku Sl. 5. i vidimo da model radi izvrsno, kao što je očekivano.

Slika na kojoj se prikazuje vozilo, Kopneno vozilo, vanjski, zgrada

Opis je automatski generiran

Sl. 5.7: Identifikacija automobila korištenjem unaprijed treniranog modela [15]

S druge strane, korištenjem modela treniranog na vlastito odabranom skupu podataka dobivamo sliku Sl. 5. i vidimo da model ne radi baš najbolje, odnosno prepoznaje samo automobil u prednjem planu.

Slika na kojoj se prikazuje vozilo, Kopneno vozilo, vanjski, zgrada

Opis je automatski generiran

Sl. 5.8: Identifikacija automobila korištenjem vlastito treniranog modela [15]

Zatim pogledajmo kako radi brojanje vozila na video snimci. Na slici Sl. 5. nalazi se jedan isječak iz videosnimke [21] na kojemu se vozila na slici dijele u dvije kategorije: vozila koja ulaze, odnosno idu „prema nama“, i vozila koja izlaze, odnosno udaljavaju se iz naše perspektive. Pred kraj snimke prebrojani su 3 ulazna i 3 izlazna automobila te 2 izlazna kamiona. Granica koja prati prolaz vozila nalazi se nešto ispod sredine slike kako bi se povećala vjerojatnost za točnim prebrojavanjem budući da će model lakše detektirati veće, odnosno bliže objekte.

Slika na kojoj se prikazuje vanjski, vozilo, cesta, Kopneno vozilo

Opis je automatski generiran

Sl. 5.9: Brojanje vozila na video snimci korištenjem unaprijed treniranog modela

Rezultati na vlastito treniranom modelu nalaze se na slici Sl. 5.1 i su izbrojana su samo dva vozila, odnosno jedan ulazni i jedan izlazni automobil, dakle drugi model ima znatno lošiji performans prilikom brojanja vozila u odnosu na prvi model.

Slika na kojoj se prikazuje cesta, vanjski, put, scena

Opis je automatski generiran

Sl. 5.10: Brojanje vozila na video snimci korištenjem vlastito treniranog modela

Uzimajući u obzir statistike i eksperimente, možemo zaključiti kako unaprijed trenirani model ima znatno kvalitetniji i konzistentniji performans od vlastito treniranog modela. Međutim, to je očekivano, budući da bi za kvalitetniji model trebali puno više podataka za učenje i trebali bi eksperimentirati s idealnim brojem epoha (npr. 150 ili 200).

# **Zaključak**

Na kraju rada piše se kratak zaključak, duljine do najviše jedne stranice.

# **Literatura**

1. Google Cloud – What is Aritificial Intelligence? Poveznica: <https://cloud.google.com/learn/what-is-artificial-intelligence>; 31. svibnja 2024.
2. Bojana Dalbelo Bašić, Marko Čupić, Jan Šnajder, Uvod u umjetnu inteligenciju: Strojno učenje, prezentacija, 2024.
3. AWS – What is Deep Learning? Poveznica: <https://aws.amazon.com/what-is/deep-learning/>; 31. svibnja 2024.
4. Azure – What is Computer Vision? Poveznica: <https://azure.microsoft.com/en-us/resources/cloud-computing-dictionary/what-is-computer-vision#object-classification>; 1. lipnja 2024.
5. Computer Vision – What it is and why it matters. Poveznica: <https://www.sas.com/en_us/insights/analytics/computer-vision.html>; 1. lipnja 2024.
6. Bojana Dalbelo Bašić, Marko Čupić, Jan Šnajder, Uvod u umjetnu inteligenciju: Umjetne neuronske mreže, prezentacija, 2024.
7. The Basics of Neural Networks. Poveznica: <https://towardsdatascience.com/the-basics-of-neural-networks-neural-network-series-part-1-4419e343b2b>; 1. lipnja 2024.
8. What Is a Learning Curve in Machine Learning? Poveznica: <https://www.baeldung.com/cs/learning-curve-ml>; 2. lipnja 2024.
9. A Guide to Convolutional Neural Networks. Poveznica: <https://saturncloud.io/blog/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way/>; 2. lipnja 2024.
10. Tutorialspoint – Prewitt operator. Poveznica: <https://www.tutorialspoint.com/dip/prewitt_operator.htm>; 2. lipnja 2024.
11. GeeksForGeeks - Introduction to Convolution Neural Network. Poveznica: <https://www.geeksforgeeks.org/introduction-convolution-neural-network/>; 2. lipnja 2024.
12. Robofolow – What is YOLO? Poveznica: <https://blog.roboflow.com/guide-to-yolo-models/>; 3. lipnja 2024.
13. Roboflow – What is R-CNN? Poveznica: <https://blog.roboflow.com/what-is-r-cnn/>; 3. lipnja 2024.
14. Roboflow – What is YOLOv8? Poveznica: <https://blog.roboflow.com/whats-new-in-yolov8/>; 3. lipnja 2024.
15. Pexels - Slika automobila. Poveznica: <https://www.pexels.com/photo/cars-parked-near-buildings-during-daytime-92866/>; 5. lipnja 2024.
16. CVAT – Open Dana Annotation Platform. Poveznica: <https://www.cvat.ai/>; 4. lipnja 2024.
17. Roboflow. Poveznica: <https://roboflow.com/>, 4. lipnja 2024.
18. VehicleDetection-YOLOv8 – kaggle. Poveznica: <https://www.kaggle.com/datasets/alkanerturan/vehicledetection?select=VehiclesDetectionDataset>; 12. svibnja 2024.
19. Classification: Precision and Recall. Poveznica: <https://developers.google.com/machine-learning/crash-course/classification/precision-and-recall>; 5. lipnja 2024.
20. Unsplash: City street with cars photo. Poveznica: <https://unsplash.com/photos/city-street-with-cars-during-daytime-kzsMak9p9L0>; 6. lipnja 2024.
21. Roboflow: Vehicle Counting Video. Poveznica: <https://docs.google.com/uc?export=download&confirm=CONFIRM_TOKEN&id=1pz68D1Gsx80MoPg-_q-IbEdESEmyVLm->; 6. lipnja 2024.
22. What is Flask? Poveznica: <https://www.bairesdev.com/blog/what-is-flask/>; 10. lipnja 2024.

# **Sažetak**

# **Summary**