



UNIVERZITET U SARAJEVU
ELEKTROTEHNIČKI FAKULTET



Prepoznavanje saobraćajnih znakova

PROJEKTNI ZADATAK

Prepoznavanje oblika i obrada slike

Tim Casper

Članovi tima:

Berin Mašović

Elhattab Yahia Aissa

Asmir Prašović

Nedim Džindo

Amir Hastor

Sadržaj

1. Ciljevi projekta.....	1
1.1. Ograničenja problema.....	1
2. Referentni radovi.....	3
2.1. Rad 1 - Stallkamp et al. (2011) - GTSRB Benchmark.....	3
2.2. Rad 2 - Traffic Sign Detection with Machine Learning and Artificial Intelligence.....	6
2.3. Rad 3 - Traffic Signs Detection and Recognition System using Deep Learning.....	8
2.4. Rad 4 - Traffic Sign Recognition Using YOLOv5 and Transfer Learning.....	11
2.5. Rad 5 - Improved Traffic Sign Detection and Recognition Algorithm for Intelligent Vehicles.....	14
3. Profiliranje podataka.....	17
3.1. Tipovi podataka.....	19
3.2. Rasподjela podataka.....	20
4. Plan implementacije.....	22
4.1. Definicija problema.....	22
4.2. Plan rješenja i koraci izrade projekta.....	23
4.2.1. Korak 1 – Analiza problema i prikupljanje podataka.....	23
4.2.2. Korak 2 – Predobrada podataka.....	23
4.2.3. Korak 3 – Segmentacija slike.....	23
4.2.4. Korak 4 – Ekstrakcija karakteristika.....	23
4.2.5. Korak 5 – Odabir modela i treniranje.....	23
4.2.6. Korak 6 – Validacija i evaluacija modela.....	24
4.2.7. Korak 7 – Implementacija sistema.....	24
5. Predprocesiranje skupa podataka.....	24
5.1. Učitavanje skupa podataka.....	24
5.2. Čišćenje podataka.....	25
5.3. Transformacija podataka.....	27
5.4. Smanjenje skupa podataka.....	27
6. Model neuronske mreže.....	28
6.1. Odabir arhitekture.....	28
6.2. Struktura neuronske mreže.....	28
6.3. Aktivacijske funkcije i inicijalizacija.....	28
6.4. Funkcija gubitka i optimizator.....	29
7. Rad sa skupom podataka.....	29
7.1. Učitavanje skupa podataka.....	29
7.2. Reduciranje trening skupa.....	29
7.3. Podjela na trening, validacijski i testni skup.....	29
8. Evaluacija modela.....	29
8.1. Metrike performansi.....	29

8.1.1. Eksperimentisanje sa veličinom slike.....	31
8.2. Konfuzijska matrica.....	32
8.3. Vizualni prikaz predikcija.....	33
9. Implementacija grafičkog korisničkog interfejsa.....	34
9.1 Opis zadatka.....	34
9.2 Način 1: Google Colab.....	34
9.2.1 Korištene biblioteke.....	34
9.2.2 Predobrada slike.....	34
9.2.3 Učitavanje slike.....	35
9.2.4 Predikcija pomoću modela.....	35
9.2.5 Izlaz programa.....	35
9.2.6 Zaključak.....	35
9.3 Način 2: Visual Studio Code – GUI aplikacija.....	35
9.3.1 Cilj GUI aplikacije.....	37
9.3.2 Korištene biblioteke.....	38
9.3.3 Podešavanje stranice (UI).....	38
9.3.4 Model i konfiguracija.....	38
9.3.5 Mapa klasa (GTSRB).....	38
9.3.6 Predobrada slike.....	38
9.3.7 Upload i prikaz slike.....	39
9.3.8 Predikcija i prikaz rezultata.....	39
9.3.9 Referentna slika (Meta folder).....	39
9.3.10 Top-3 predikcije.....	39
9.3.11 Zaključak.....	40
9.4 Razlika između Colab i GUI.....	40
Lista slika.....	41
Lista tabela.....	42

1. Ciljevi projekta

1. Identifikacija i prepoznavanje saobraćajnih znakova

- Razviti model koji precizno detektuje i klasificuje saobraćajne znakove sa ulaznih slika.
- Omogućiti razlikovanje tipova znakova (zabrane, upozorenja, obaveze, informacija).

2. Priprema i obrada slika

- Normalizovati sve slike na istu rezoluciju, ali pritom da ostane dovoljno informacija.
- Primijeniti filtriranje nad slikama u cilju poboljšanja tačnosti prepoznavanja.

3. Ekstrakcija karakteristika

- Identifikacija ključnih karakteristika svakog saobraćajnog znaka, poput boje, oblika, simbola ili teksta.
- Osigurati da ekstrakcija bude otporna na promjene osvjetljenja i perspektive.

4. Odabir i treniranje modela

- Izbor odgovarajućeg algoritma mašinskog učenja (npr. neuronske mreže, SVM) za prepoznavanje saobraćajnih znakova.
- Priprema i segmentacija podataka za trening i testiranje modela.

5. Validacija i evaluacija modela

- Testiranje modela na skupu podataka koji obuhvata različite uslove osvjetljenja, perspektive i pozadinske elemente.
- Evaluacija performansi modela koristeći metrike poput tačnosti, preciznosti i odziva.
- Analizirati pogrešno klasifikovane primjere i potencijalne uzroke grešaka.

6. Implementacija sistema

- Integracija razvijenog modela u stvarni sistem za prepoznavanje saobraćajnih znakova.
- Omogućiti učitavanje slike i prikaz prepoznatog znaka.

1.1. Ograničenja problema

1. Tehnička ograničenja

- Potrebni hardverski resursi za treniranje CNN modela (GPU preporučen).
- Moguće sporije izvršavanje kod obrade slika u realnom vremenu.

2. Funkcionalna ograničenja

- Model može imati poteškoće s prepoznavanjem znakova u lošim uslovima (zamagljene slike, sjene, promjene boje).
- Ograničeno prepoznavanje znakova koji ne pripadaju GTSRB datasetu.

3. Organizaciona i vremenska ograničenja

- Projekat se mora završiti u okviru nastavnog plana, pa je vremenski okvir ograničen.
- Moguće je da neće biti vremena za razvoj kompletne real-time verzije.

2. Referentni radovi

2.1. Rad 1 - Stallkamp et al. (2011) - GTSRB Benchmark

Johannes Stallkamp, Marc Schlipsing, Jan Salmen, 2011

Sažetak

Ovaj rad predstavlja i formalno uvodi GTSRB dataset, koji je postao standard u istraživanju prepoznavanja saobraćajnih znakova. Autori su prikazali različite izazove kao što su promjena osvjetljenja, ugao posmatranja, zamućenje slike i zaklanjanje objekata. Cilj rada bio je pružiti realan i sveobuhvatan skup podataka za treniranje i testiranje algoritama mašinskog učenja.

Korištene metode

1) Priprema i anotacija skupa podataka

Autori su kreirali veliki skup realnih fotografija saobraćajnih znakova snimljenih u stvarnim uslovima vožnje. Svaka slika je ručno anotirana, što znači da je svakom znaku dodijeljena tačna klasa i okvir (bounding box ako je znak bio u originalnoj sceni).

2) Ekstrakcija vizuelnih karakteristika

HOG (Histogram of Oriented Gradients)

Ova metoda mjeri kako se smjer ivica na slici mijenja. Saobraćajni znakovi imaju jake i jasne oblike (krug, trougao, kvadrat, simboli), zato HOG dobro hvata strukturu znakova.

- Ustanovljava oblike i konture
- Zanemaruje boju i fokusira se na linije/ivice

Color-based segmentation

Boje imaju ključnu ulogu, zato se koristila kolor segmentacija da se olakša pronalaženje znaka.

- Prepoznavanje crvenih obruča
- Prepoznavanje plavih pozadina
- Maskiranje slikovnih regiona koji ne odgovaraju bojama znakova

3) Klasifikacioni modeli

Nakon izdvajanja osobina, koristili su više modela mašinskog učenja radi poređenja performansi:

Metoda	Opis
SVM (Support Vector Machine)	Standardni algoritam za klasifikaciju slika - jako precizan na manjim datasetima
Random Forest	Ensemble metoda - koristi više stabala za robusnije predviđanje
KNN (k-nearest neighbors)	Jednostavan i spor, ali koristan za poređenje
Boosting method	Kombinovanje slabijih klasifikatora da bi se dobio jači

Tabela 1: Opis i prikaz korištenih metoda

4) Evaluacija i benchmark sistem

Autori su definisali standardni način procjene performansi:

- Podjela podataka: trening / test
- Mjerenje tačnosti za svaku od 43 klase
- Analiza grešaka - posebno za slične znakove (npr. Ograničenja brzine)

5) Eksperimentalno poređenje sa CNN pristupom (rani CNN model)

CNN pristup je uključivao:

- Konvolucione slojeve za automatsku ekstrakciju karakteristika
- Trening bez potrebe za ručnim dizajniranjem feature-a

Najvažnije tehnike

1) CNN + MLP (IDSIA)

Korištena kombinacija konvolucijskih neuronskih mreža i višeslojnog perceptron-a. Slike su normalizovane, izrezane na 48×48 piksela i obrađene kroz max-pooling slojeve radi bolje otpornosti..

2) ConvNets (sermanet)

Upotrebljena C++ implementacija konvolucijskih mreža (EBLearn), trenirana na obojenim slikama 32×32 piksela uz pun nadzor, postigla 98,97 % tačnosti.

3) HOG + LDA + VQ (noob)

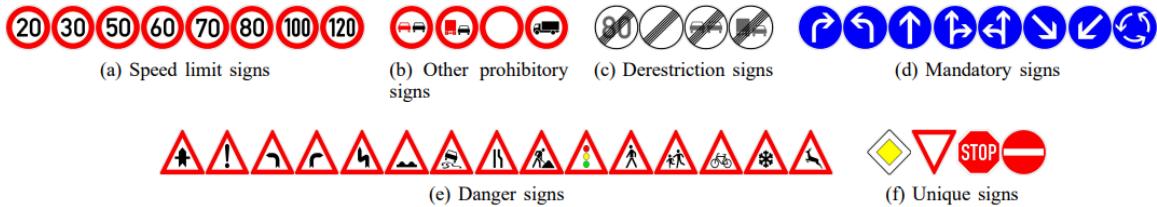
Kombinacija histograma orijentisanih gradijenata, linearne diskriminantne analize i vektorske kvantizacije radi poboljšanja tačnosti i brzine prepoznavanja znakova.

Rezultati

- Najbolji klasični modeli: ~98.98% tačnosti
- CNN modeli (u to vrijeme, istraživački): ~99.15% tačnosti

Prednosti

- Prvi standardizovan dataset za prepoznavanje saobraćajnih znakova
- Velik broj klasa (43 klase)
- Slike su snimane u stvarnom saobraćaju, a ne laboratorijski
- Analiza klasičnih metoda i rani CNN pristup
- Dataset je besplatan i javno dostupan



Slika 1: Prikaz saobraćajnih znakova

Nedostaci

- Dataset sadrži znakove isključivo iz Njemačke.
- GTSRB daje već isječene slike znakova, tj. znak je već izdvojen iz scene. Dataset je odličan za klasifikaciju, ali nije kompletan za real-time aplikacije.
- Nema video sekvenci
- Ne pokriva ekstremne situacije vrlo deformisane / polomljene znakove

2.2. Rad 2 - Traffic Sign Detection with Machine Learning and Artificial Intelligence

Darshan K. Koli, Vaibhav N. Rajput, Sachin S. Badgujar, Sujitkumar V. Chaudhari, 2025

Sažetak

Rad obrađuje sistem za prepoznavanje saobraćajnih znakova koristeći konvolucione neuronske mreže (CNN). Cilj je razviti precizan i brz model koji može u stvarnom vremenu prepoznati 43 klase znakova sa GTSRB (German Traffic Sign Recognition Benchmark) skupa podataka. Model je implementiran u PyTorchu, a za realnu primjenu korišten je OpenCV i kamera. Postignuta je tačnost iznad 99%, što potvrđuje efikasnost predloženog pristupa.

Korištene metode

1) Skup podataka:

GTSRB, sa više od 35.000 označenih slika u tom periodu i preko 40 klasa.

2) Predobrada:

- Promjena veličine slika na 64×64 piksela.
- Normalizacija na osnovu ImageNet vrijednosti.
- Augmentacija: rotacije, affine transformacije, promjene boje.

3) Model:

- Tri konvolucijska bloka (Conv2D + BatchNorm + ReLU + MaxPooling).
- Fully connected sloj sa dropoutom (0.3) i izlazni sloj (Softmax za 43 klase).

4) Trening:

- Framework: PyTorch
- Optimizator: Adam ($\text{lr} = 0.001$)
- Gubitak: CrossEntropyLoss
- Epochs: 5, Batch size: 64
- Korištenje GPU-a (CUDA) ako je dostupan.

5) Evaluacija:

Validacijska tačnost, gubitak kroz epohe, FPS performanse u realnom vremenu

Najvažnije tehnike

1) Konvolucione neuronske mreže (CNN)

CNN automatski uče karakteristike slike kroz više slojeva filtriranja, bez ručne ekstrakcije. Ova tehnika omogućava visoku tačnost i prepoznavanje složenih vizualnih uzoraka znakova. Prepoznaće važne karakteristike na slici kao što su ivice, oblici i obrasci, kroz niz konvolucijskih slojeva.

Konvolucija je matematička operacija koja se u CNN-u koristi za izdvajanje karakteristika iz slike. Radi tako što mali filter (kernel) prolazi kroz sliku i računa proizvode piksela i vrijednosti filtera. Rezultat su nove slike (tzv. feature mape) koje ističu važne detalje koje mreža kasnije koristi za prepoznavanje objekata.

2) Data augmentation

Korištene su rotacije, translacije i promjene boje kako bi se simulirali različiti uslovi snimanja. Time se povećava otpornost modela i smanjuje prenaučenost (overfitting).

3) Batch Normalization i Dropout

BatchNorm stabilizuje trenin normalizacijom vrijednosti između slojeva, a Dropout isključuje dio neurona radi bolje generalizacije. Zajedno poboljšavaju tačnost i smanjuju rizik od prenaučenosti.

4) PyTorch framework

PyTorch omogućava efikasno treniranje modela na GPU-u i jednostavnu implementaciju CNN arhitekture. Koristi se i za kontrolu epoha, gubitka i evaluaciju rezultata.

5) OpenCV integracija

OpenCV služi za obradu slike u realnom vremenu - detekciju znakova, izdvajanje regija i prikaz rezultata s kamere. Omogućava praktičnu primjenu modela u stvarnim uslovima.

6) Learning Rate Scheduler

ReduceLROnPlateau automatski smanjuje brzinu učenja kada validacijski gubitak stagnira. Pomaže modelu da postigne stabilniju i precizniju konvergenciju.

Rezultati

- Validacijska tačnost: >99%
- Stabilan pad trenin i validacijskog gubitka (bez overfittinga).
- Sistem u realnom vremenu radi brzinom od oko 10 FPS, zavisno od hardvera.
- Uspješno prepoznavanje svih klasa znakova nakon balansiranja podataka i augmentacije.

Prednosti

- Visoka tačnost i brza obrada u stvarnom vremenu.
- CNN arhitektura - jednostavna za implementaciju i treniranje.
- Bez ručne ekstrakcije karakteristika, model sam uči relevantne obrasce.
- Jednostavna integracija u ADAS* i autonomne sisteme vozila.

Nedostaci

- Zavisnost od kvaliteta slike - slabije performanse pri lošem osvjetljenju, kiši ili magli.
- Neuravnotežene klase u GTSRB-u mogu uticati na tačnost rijetkih znakova.
- Model radi odlično na GPU-u, ali teško se prenosi na mobilne/ugrađene uređaje bez kompresije.
- Nedostatak video-podataka ograničava sposobnost modela da uči vremenske obrasce.

2.3. Rad 3 - Traffic Signs Detection and Recognition System using Deep Learning

Pavly Salah Zaki, Marco Magdy William, Bolis Karam Soliman, Kerolos Gamal Alexsan, Maher Mansour, Magdy El-Moursy, Kerolos Khalil, 2020

Sažetak

U radu Traffic Signs Detection and Recognition System using Deep Learning autori predstavljaju sistem za automatsko prepoznavanje saobraćajnih znakova koji koristi metode dubokog učenja. Cilj istraživanja bio je razviti efikasan i precizan model sposoban za rad u realnom vremenu, uzimajući u obzir različite izazove poput promjene osvjetljenja, vremenskih uslova i vidljivosti.

Rad se fokusira na dvije arhitekture – Faster R-CNN Inception v2 i Tiny-YOLOv2, koje su trenirane i testirane na German Traffic Sign Detection Benchmark (GTSDB) skupu podataka.

Rezultati pokazuju da je F-RCNN Inception v2 postigao 96% tačnost, dok je Tiny-YOLOv2, iako manje precizan (73%), omogućio značajno veću brzinu obrade, što ga čini pogodnim za ugradbene sisteme poput Raspberry Pi 3.

Autori zaključuju da se kombinacijom transfer učenja i konvolucijskih neuronskih mreža (CNN) može postići visok nivo tačnosti i efikasnosti, što ovaj pristup čini pogodnim za integraciju u ADAS (Advanced Driver Assistance Systems).

Korištene metode

U radu su korištene dvije napredne metode detekcije objekata zasnovane na dubokim konvolucijskim neuronskim mrežama:

1) Faster R-CNN (Region-Based Convolutional Neural Network)

- Kombinuje Region Proposal Network (RPN) i klasifikator zasnovan na CNN-u.
- Koristi Inception v2 arhitekturu kao bazni model, optimiziranu 1x1 i 3x3 konvolucijskim slojevima.
- Trening je izведен pomoću tehnike fine-tuninga, gdje je model pretrenirani na velikim skupovima podataka (npr. COCO) dodatno prilagođen GTSDB skupu.

2) Tiny-YOLOv2 (You Only Look Once – Lightweight Version)

- Model jedinstvenog prolaza koji obrađuje slike u realnom vremenu.
- Prilagođen za uređaje s ograničenim resursima (bez GPU-a).
- Trening je uključivao optimizaciju hiperparametara, smanjenje broja slojeva i upotrebu 1x1 konvolucija radi ubrzanja.

Oba modela su trenirana na četiri kategorije znakova: Prohibitory, Mandatory, Danger i Stop, s korištenjem RGB i HSV konverzije slika. Evaluacija je obavljena na više platformi – od računara s GPU-om (GTX 1070) do ugradbenih uređaja poput Raspberry Pi 3 Model B+.

Najvažnije tehnike

1) Transfer learning:

Iskorištenje već treniranih mreža radi postizanja boljih rezultata na manjem skupu podataka.

2) Fine-tuning

Ponovni trening modela uz prilagođavanje hiperparametara poput stope učenja, regularizacije i drop-outa.

3) Data augmentation

Povećanje broja uzoraka promjenom orijentacije, osvjetljenja i kontrasta slika.

4) Non-maximum suppression

Odabir najvjerojatnijih regija objekata kako bi se smanjile greške u detekciji.

5) Intersection over Union (IoU):

Metrika za procjenu tačnosti predikcije položaja objekta.

6) HSV konverzija

Poboljšanje detekcije boja pod različitim svjetlosnim uslovima.

Rezultati

Model	Tačnost (mAP)	Brzina (FPS)
F-RCNN Inception v2	96%	~25
Tiny-YOLOv2	73%	~70
SSD MobileNet v2	83%	~42

Tabela 2: Prikaz rezultata korištenih metoda

- F-RCNN Inception v2 pruža najvišu tačnost i pouzdanost, ali ima veću računski zahtjevnost.
- Tiny-YOLOv2 ima znatno veću brzinu, što ga čini pogodnim za real-time aplikacije.
- Na Raspberry Pi 3, Tiny-YOLOv2 je postigao 7 FPS, što je dovoljno za osnovnu implementaciju u sistemima za asistenciju vozaču.
- U simulacijama (TASS PreScan), F-RCNN je postigao stabilan rad pri 20 FPS uz visoku preciznost prepoznavanja znakova.

Prednosti

- Visoka tačnost prepoznavanja čak i u složenim uslovima (magla, sjenke, različiti uglovi).
- Mogućnost rada u realnom vremenu, posebno kod Tiny-YOLOv2 modela.
- Skalabilnost i prenosivost – modeli se mogu implementirati na različitim uređajima, od snažnih GPU računara do malih ugradbenih sistema.
- Primjenjivost u ADAS sistemima, što doprinosi sigurnosti u saobraćaju.
- Efikasna upotreba transfer učenja, čime se smanjuje potreba za ogromnim skupovima podataka.

Nedostaci

- Ograničen skup podataka (GTSDB) – sadrži samo 900 slika, što nije dovoljno za sve kategorije znakova.
- Nedostatak uravnoteženosti klasa – neke kategorije imaju vrlo malo uzoraka (<20).
- F-RNN model prezahtjevan za rad na slabijem hardveru; nije mogao biti pokrenut na Raspberry Pi.
- Tiny-YOLOv2 ima nižu tačnost, što može uzrokovati greške u prepoznavanju znakova sličnog izgleda.
- Zavisnost od kvaliteta osvjetljenja i ugla kamere, što može uticati na rezultate u stvarnim uslovima.

2.4. Rad 4 - Traffic Sign Recognition Using YOLOv5 and Transfer Learning

Omran Nacir, Amna Maraoui, Imen Werda, Belgacem Hamd, 2022

Sažetak

U ovom radu autori su predstavili sistem za detekciju i prepoznavanje saobraćajnih znakova koristeći YOLOv5 — modernu arhitekturu dubokog učenja zasnovanu na konvolucijskim nevronskim mrežama. Cilj istraživanja bio je razviti model koji može u realnom vremenu prepoznati više klase znakova iz GTSRB (German Traffic Sign Recognition Benchmark) skupa podataka, uz visoku tačnost i efikasnost obrade. Model je treniran pomoću transfer učenja na unaprijed istreniranom COCO skupu, a zatim fino podešen (fine-tuning) za GTSRB, čime je postignuta visoka tačnost uz minimalne resurse i kratko vrijeme treniranja.

Korištene metode

1) Dataset i priprema podataka

- Korišten je GTSRB skup podataka (43 klase saobraćajnih znakova).
- Slike su skalirane na 640×640 piksela, normalizirane i augmentirane (rotacije, horizontalni flip, promjena svjetline i kontrasta).
- Dataset je podijeljen u 80% za trening i 20% za testiranje.



Slika 2: Primjeri slika iz GTSRB skupa podataka koji sadrži 43 klase saobraćajnih znakova.

2) YOLOv5 arhitektura

- Model se sastoji od tri glavna dijela: Backbone, Neck i Head.
 - **Backbone** (CSPDarknet53): ekstrakcija karakteristika slike.
 - **Neck** (PANet): kombinacija višeslojnih karakteristika za bolju detekciju manjih objekata.
 - **Head**: generiše bounding box koordinate i vjerovatnoću klase.
- YOLOv5 koristi tzv. one-stage detection, što znači da detektuje i klasificiše objekte u jednom prolazu slike kroz mrežu.



Slika 3: Struktura YOLOv5 modela i način prenosa informacija između slojeva.

3) Trening i optimizacija

- Korištena je tehnika transfer learning – početno stanje mreže preuzeto s COCO dataset-a.
- Optimizator: Adam, stopa učenja 0.001.
- Loss funkcija: kombinacija GIoU Loss (za lokaciju), Objectness Loss i Class Loss.
- Trening je izведен kroz 100 epoha na GPU-u.

4) Evaluacija

- Metrike: mAP@0.5 (mean Average Precision), preciznost, odziv (recall) i FPS (frame per second).
- Evaluacija je rađena na test skupu GTSRB podataka.

Najvažnije tehnike

1) Transfer Learning

Korištenje već naučenih težina s COCO dataset-a radi bržeg konvergiranja i boljih rezultata na manjem skupu podataka.

2) Data Augmentation

Simuliranje različitih uslova snimanja (promjene svjetline, rotacije, zamućenja).

3) Non-Maximum Suppression (NMS)

Uklanjanje preklapajućih detekcija istog objekta.

4) Batch Normalization i Leaky ReLU

Stabilizacija treniranja i poboljšanje generalizacije.

5) Anchor Boxes optimizacija

Prilagođavanje veličina anchor okvira veličini znakova u GTSRB skupu.

Rezultati

- mAP@0.5: 98.7%
- Preciznost: 99.1%
- Odziv (Recall): 98.4%

- Brzina obrade: 60 FPS na GPU (RTX 3060)

Model je pokazao da može u realnom vremenu pouzdano prepoznavati sve klase znakova, uz vrlo visoku tačnost. Također, performanse su ostale stabilne i pri lošijem osvjetljenju, zahvaljujući augmentaciji i optimizaciji anchor box-ova.

Prednosti

- Visoka tačnost i brzina — omogućava detekciju i klasifikaciju u realnom vremenu.
- Jednostavna implementacija i trening uz transfer learning.
- Mala veličina modela (manje od 30 MB) omogućava upotrebu na ugradbenim uređajima (npr. Raspberry Pi, Jetson Nano).
- Sposobnost detekcije više znakova na jednoj slici.
- Robusnost na promjene svjetline, rotacije i djelimične zaklonjenosti.

Nedostaci

- Ovisnost o kvaliteti ulazne slike – magla, kiša i jaka svjetlost mogu smanjiti tačnost.
- Model zahtijeva GPU za treniranje; izvođenje na CPU-u je sporo.
- Nije optimiziran za video sekvence (nedostatak vremenske analize).
- Potrebno ručno podešavanje anchor box-ova za maksimalne rezultate.

2.5. Rad 5 - Improved Traffic Sign Detection and Recognition Algorithm for Intelligent Vehicles

Jingwei Cao, Chuanxue Song, Silun Peng, Feng Xiao, Shixin Song, 2019

Sažetak

Rad se bavi algoritmima, detekcijom i prepoznavanjem saobraćajnih znakova za pametna auta. Cilj je da se postigne visoka tačnost i brzina obrade prilikom prepoznavanja znakova, čak i u izazovnim okolnostima kao što su promjene osvjetljenja, djelimična zaklonjenost znakova, razne vremenske prilike i različiti uglovi kamere. Da bi se to postiglo, autori su predložili dvostepeni sistem koji se sastoji od faze detekcije i faze klasifikacije. U fazi detekcije, korišten je HSV prostorni model boja u kombinaciji sa threshold segmentacijom kako bi se izdvojile regije slike koje sadrže crvene, plave ili žute tonove — boje karakteristične za saobraćajne znakove. Ovaj pristup omogućava da se područje od interesa brzo izdvoji iz kompleksne scene bez potrebe za dubokim modelima u ovoj fazi, čime se smanjuje ukupna računarska složenost, dok je u fazi prepoznavanja implementirana konvolucijska neuronska mreža (CNN).

Korištene metode

1) U fazi detekcije

Koriste se „HSV boja prostor + prostorna threshold segmentacija“ da se izdvoje potencijalni znakovi na slici.

2) U fazi prepoznavanja

Početni CNN je baziran na klasičnom LeNet-5 modelu, ali je unaprijeđen: uvodi se Gabor kernelka kao početni konvolucionisani kernel, dodaje se batch normalization poslije pooling slojeva, kao i optimizator Adam,

3) Dataset

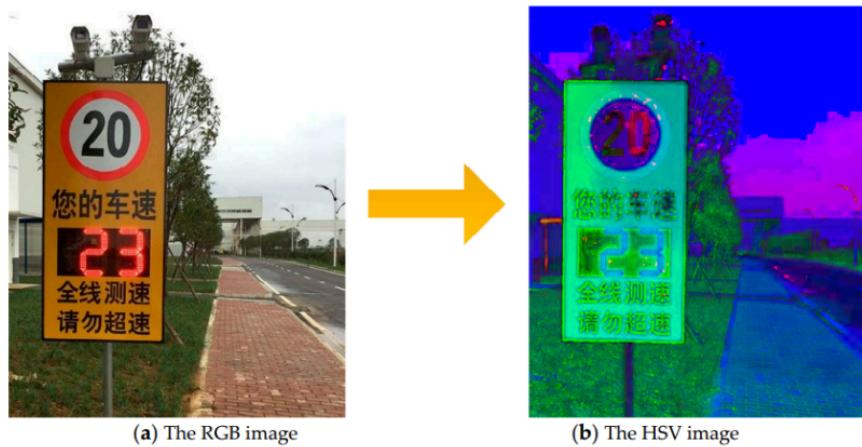
Koristi se GTSRB za klasifikaciju/prepoznavanje.

Najvažnije tehnike

1) Korištenje HSV prostora boja za detekciju znakova

Umjesto RGB modela, u fazi detekcije koristi se HSV (Hue–Saturation–Value) prostor boja koji bolje odražava ljudsku percepciju boja.

- Na osnovu odabira raspona nijansi (Hue) izdvajaju se oblasti slike koje sadrže tipične boje saobraćajnih znakova – crvenu, plavu i žutu.
- Ova metoda omogućava brzu i preciznu segmentaciju znakova čak i pri promjenama osvjetljenja, čime se smanjuje broj nepotrebnih regija koje kasnije obrađuje neuronska mreža.



Slika 4: Konverzija RGB slike u HSV sliku

2) Threshold segmentacija

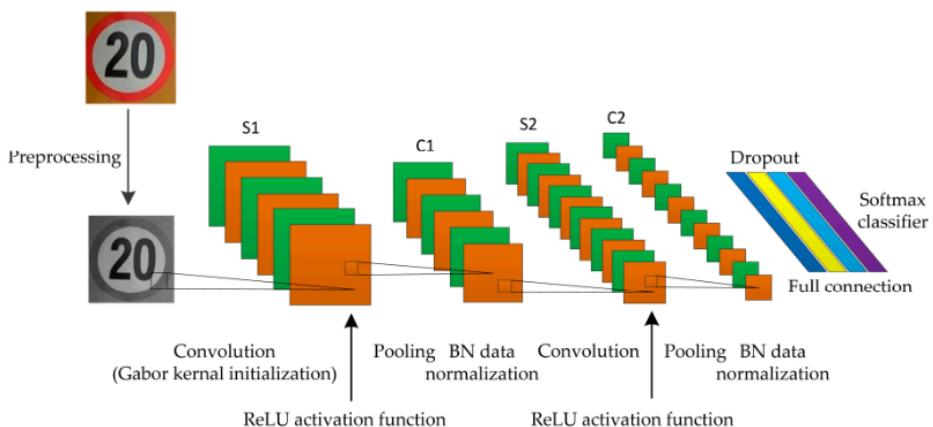
Nakon prelaska slike u HSV prostor, primjenjuje se thresholding (pragovna segmentacija).

- Ovim pristupom se jasno razdvajaju znakovi od pozadine, što znatno pojednostavljuje dalju obradu i smanjuje količinu šuma.
- Rezultat ove faze je skup „kandidata“ koji sadrže moguće znakove.

3) Konvolucijska neuronska mreža (CNN)

Za klasifikaciju znakova koristi se CNN model, jer automatski uči prostorne i vizuelne obrazce bez potrebe za ručnim definisanjem karakteristika.

- Osnovna arhitektura se zasniva na LeNet-5 modelu, ali je proširena i optimizovana:
 - Dodani su Gabor filteri u prvim slojevima radi boljeg hvatanja ivica i orijentacija.
 - Primijenjena je Batch Normalization tehnika za stabilizaciju i ubrzanje treniranja.
 - Uveden je Dropout sloj kako bi se spriječila prenaučenost i povećala generalizacija.
 - Aktivacijska funkcija ReLU koristi se za nelinearnu transformaciju podataka i smanjenje problema zasićenja neurona.



Slika 5: Poboljšana verzija LuNet-5 modela

4) Gabor filteri

Gabor filteri su korišteni kao inicijalni konvolucijski filteri u prvim slojevima CNN-a. Ovi filtri su posebno pogodni za prepoznavanje orijentisanih ivica i tekstura, što je ključno za tačno razlikovanje oblika i simbola na saobraćajnim znacima. Korištenje Gabor filtera omogućava modelu da već u ranim fazama ekstrahuje značajne prostorne karakteristike.

5) Batch Normalization

- Primjenjuje se nakon svakog pooling sloja radi normalizacije aktivacija unutar mreže.
- Ova tehnika stabilizuje proces treniranja, omogućava veću brzinu konvergencije i smanjuje ovisnost o inicijalnim parametrima.

Rezultati



Slika 6: Grupacija saobraćajnih znakova

Sequence Number	Traffic Signs	Test Images	TP	FN	Accurate Recognition Rate (%)	Average Processing Time (ms)/Frame
1	Speed Limit	1000	997	3	99.70	5.tra
2	Danger	1000	999	1	99.90	5.kol
3	Mandatory	1000	997	3	99.70	5.vlj
4	Prohibitory	1000	998	2	99.80	4.ruj
5	Derestriction	1000	994	6	99.40	6.tra
6	Unique	1000	1000	0	100.00	4.srp
Total	-	6000	5985	15	99.75	5.tra

Tabela 3: Prikaz rezultata klasifikacije

Prednosti

- Predloženi algoritam postiže 99,75% tačnosti, nadmašujući većinu usporednih metoda, uključujući Multi-CNN i HOG+PCA.
- Prosječno vrijeme obrade po slici je 5,4 ms, što omogućava integraciju u sisteme autonomne vožnje i napredne sisteme pomoći vozaču (ADAS).
- Sistem je otporan na promjene osvjetljenja, kut gledanja, djelomično zaklonjene znakove i šum u slici zahvaljujući kombinaciji HSV detekcije, data augmentation i Gabor filtera.
- Faza detekcije smanjuje broj nepotrebnih regija, čime CNN model može brže i preciznije obraditi slike.
- Korištenje Batch Normalization i Dropout slojeva smanjuje prenaučenost i omogućava modelu pouzdano prepoznavanje znakova na neviđenim podacima.
- Arhitektura i proces treniranja mogu se lako prilagoditi novim datasetovima ili dodatnim klasama saobraćajnih znakova.

Nedostaci

- Ekstremno loše osvjetljenje, niska rezolucija, zamućenje ili prevelik nagib mogu uzrokovati pogrešnu klasifikaciju.
- Algoritam je treniran na 43 klase GTSRB skupa i ne može pouzdano prepoznati znakove izvan tog skupa bez dodatnog treniranja.
- Treniranje modela zahtijeva GPU i znatno vrijeme optimizacije, iako je sama predikcija brza.
- Iako HSV prostor poboljšava detekciju, znakovi s izbljedjelim bojama ili neuobičajenim nijansama mogu biti propušteni u fazi detekcije.

3. Profiliranje podataka

Naziv skupa podataka i izvor

Za potrebe realizacije ovog projekta koristi se German Traffic Sign Recognition Benchmark (GTSRB) dataset, pronađen na platformi Kaggle.

Podaci su zapisani u CSV formatu i podijeljeni u tri dijela:

- Meta podaci
- Podaci za treniranje
- Podaci za testiranje

Opis i svrha

Skup podataka sadrži preko 50.000 slika saobraćajnih znakova, raspoređenih u 43 različite klase. Na osnovu ovih podataka moguće je kreirati računarski model koji može imati primjenu u:

- Obuci vozača
- Sistemima za navigaciju
- Autonomnim vozilima

Dakle, skup podataka može se koristiti za kreiranje modela koji će zaključiti značenje prometnog znaka na osnovu njegove slike.

Struktura skupa podataka

CSV datoteke sadrže putanje do slika koje se nalaze u posebnim folderima Meta, Train i Test koji su organizovani po klasama.

CSV datoteke uključuju:

- naziv slike
- oznaku klase (label)
- koordinate bounding boxa (Width, Height, Roi.X1, Roi.Y1, Roi.X2, Roi.Y2)

Meta podaci olakšavaju filtriranje i grupisanje slika po klasama, što je korisno za treniranje modela.

Veličina skupa podataka

Skup podataka GTSRB sadrži:

- CSV za treniranje (Train): 39.209 redova i 8 kolona
- CSV za testiranje (Test): 12.630 redova i 8 kolona
- Ukupan broj slika u skupu: 51.881

Skup podataka sadrži 43 klase saobraćajnih znakova.

Klasa	Br. slika								
0	210	9	1470	18	1200	27	240	36	390
1	2220	10	2010	19	210	28	540	37	210
2	2250	11	1320	20	360	29	270	38	2070
3	1410	12	2100	21	330	30	450	39	300
4	1980	13	2160	22	390	31	780	40	360
5	1860	14	780	23	510	32	240	41	240
6	420	15	630	24	270	33	689	42	240
7	1440	16	420	25	1500	34	420		
8	1410	17	1110	26	600	35	1200		

Tabela 4: Broj slika po klasama u trening skupu

Klasa	Br. slika								
0	60	9	480	18	390	27	60	36	120
1	720	10	660	19	60	28	150	37	60
2	750	11	420	20	90	29	90	38	690
3	450	12	690	21	90	30	150	39	90
4	660	13	720	22	120	31	270	40	90
5	630	14	270	23	150	32	60	41	60
6	150	15	210	24	90	33	210	42	90
7	450	16	150	25	480	34	120		
8	450	17	360	26	180	35	390		

Tabela 5: Broj slika po klasama u test skupu

3.1. Tipovi podataka

Naziv kolone	Tip podatka	Opis
Width	int64	Širina slike u pikselima
Height	int64	Visina slike u pikselima
Roi.X1	int64	X koordinata gornjeg lijevog ugla znaka
Roi.Y1	int64	Y koordinata gornjeg lijevog ugla znaka
Roi.X2	int64	X koordinata donjeg desnog ugla znaka
Roi.Y2	int64	Y koordinata donjeg desnog ugla znaka
ClassId	int64	Oznaka klase (0–42) (kategorički tip)
Path	object	Putanja do slike (tekstualni tip)

Tabela 6: Trening skup (*Train.csv*) i test skup (*Test.csv*)

Naziv kolone	Tip podatka	Opis
Path	object	Putanja do slike klase
ClassId	int64	Oznaka klase
ShapeId	int64	Oblik znaka (kategorički tip)
ColorId	int64	Boja znaka (kategorički tip)
SignId	int64	ID znaka (jedinstvena identifikacija)

Tabela 7: Meta podaci (Meta.csv)

Sve slike u trening i test skupu su u boji (RGB).

Meta podaci služe za dodatnu klasifikaciju po obliku, boji i identifikaciji znakova, što može biti korisno kod naprednog treniranja modela.

3.2. Raspodjela podataka

Kolona	Min (Train)	Median (Train)	Max (Train)	Mean (Train)	Min (Test)	Median (Test)	Max (Test)	Mean (Test)
Width	25	43	243	34	25	43	266	31
Height	25	43	225	33	25	43	232	32
Roi.X1	0	6	20	/	1	6	23	/
Roi.Y1	5	6	20	/	5	6	19	/
Roi.X2	20	38	223	/	20	38	244	/
Roi.Y2	20	38	205	/	20	38	212	/

Tabela 8: Osnovne statistike za numeričke kolone

U datasetu se pojavljuje jedna kategorička kolona - ClassId, koja predstavlja tip saobraćajnog znaka.

Skup	Min dimenzije	Max dimenzije	Prosječna širina	Prosječna visina	Tip
Train	25×25	243×225	34 px	33 px	RGB
Test	25×25	266×232	31 px	32 px	RGB

Tabela 9: Fizičke karakteristike slika

Dataset je nebalansiran, jer određene klase imaju višestruko više uzoraka od drugih. Ova nebalansiranost može negativno uticati na performanse modela, pa je preporučeno primijeniti balansiranje podataka.

Kvalitet i varijabilnost slika

Dataset se sastoji od realnih fotografija saobraćajnih znakova prikupljenih u stvarnom okruženju. Slike su snimane u različitim uslovima, što doprinosi njihovoј raznolikosti, ali istovremeno povećava složenost zadatka klasifikacije.

U skupu podataka primjetne su različite vrste varijacija:

- **Osvjetljenje:**
 - Na slikama se primjećuju varijacije u osvjetljenju
- **Ugao snimanja:**
 - Znakovi nisu uvijek prikazani frontalno
 - Dio slika je snimljen pod blagim ili većim uglovima
- **Pozadina:**
 - Može uključivati elemente poput ceste, drveća, neba ili zgrada
- **Kontrast i oštrina:**
 - Prisutan je raspon od vrlo oštih do blago zamućenih slika
 - Zavisi od udaljenosti i brzine kretanja kamere
- **Boja i zasićenje:**
 - Sve slike su u boji (RGB)
 - Mogu se uočiti manje razlike u zasićenju i balansu boja zbog različitih kamera i uslova osvjetljenja

Informacije o kontekstu

Pored osnovnih informacija o strukturi i upotrebi, važno je napomenuti da je GTSRB (German Traffic Sign Recognition Benchmark) jedan od najpoznatijih i najčešće korištenih skupova podataka u oblasti računalnog vida. Dataset se često koristi kao referentni (benchmark) skup u istraživanjima koja se bave prepoznavanjem saobraćajnih znakova, te predstavlja standard za poređenje performansi različitih modela mašinskog učenja.

Razlog za izbor skupa podataka

GTSRB skup podataka odabran je zbog svoje realnosti i veličine. Sadrži veliki broj slika saobraćajnih znakova snimljenih u različitim vremenskim i svjetlosnim uslovima, kvalitetu i položaju znakova na slici.

Ovakva raznolikost čini ga izuzetno pogodnim za projekte prepoznavanja saobraćajnih znakova, jer omogućava modelima da se bolje prilagode stvarnim uslovima. S obzirom na to da dataset sadrži dovoljno podataka za treniranje modela, to ga čini pogodnim za postizanje visokog stepena tačnosti modela.

Najbitnije informacije iz skupa podataka

Analizom skupa podataka uočeno je da, pored osnovnih karakteristika, postoje i kolone koje definišu koordinate saobraćajnog znaka na slici. Svaka slika ima specificirane koordinate gornjeg lijevog i donjeg desnog ugla znaka, što omogućava precizno određivanje njegovog položaja. Ovi podaci su posebno važni pri razvoju modela za detekciju znakova, jer doprinose tačnjem prepoznavanju njihovog položaja.

Pored koordinata, dataset sadrži i niz karakteristika koje dodatno olakšavaju proces prepoznavanja:

- **Vizuelne karakteristike znakova (oblik, boja i simbol)** – pomažu modelu u razlikovanju klase.
- **Oznaka klase (ClassId)** – definiše vrstu saobraćajnog znaka.
- **Region od interesa (ROI)** – omogućava fokusiranje na sam znak, bez pozadinskog šuma.
- **Veličina i orientacija slike** – faktori koji utiču na skaliranje i normalizaciju tokom treniranja modela.

Također, u okviru metapodataka o klasama nalaze se kolone ColorId i ShapeId, koje predstavljaju boju i oblik znaka. Ove informacije su korisne u procesu treniranja i mogu doprinijeti većoj preciznosti modela pri klasifikaciji saobraćajnih znakova.

4. Plan implementacije

4.1. Definicija problema

Problem koji projekat rješava jeste razvoj sistema koji može automatski detektovati i prepoznati saobraćajne znakove na slikama. To zahtijeva obradu slike, segmentaciju regija gdje se znakovi nalaze, ekstrakciju vizuelnih karakteristika i treniranje modela koji će klasifikovati znakove u jednu od 43 klase iz GTSRB skupa podataka. Izazovi problema uključuju različite uslove osvjetljenja, zaklonjenost znakova, varijacije u položaju i ugaonoj perspektivi te nebalansiran skup podataka.

Cilj projekta je razviti metodološki jasan, mjerljiv i efikasan sistem koji će prepoznati tip saobraćajnog znaka.

4.2. Plan rješenja i koraci izrade projekta

4.2.1. Korak 1 – Analiza problema i prikupljanje podataka

U ovom koraku analiziramo šta tačno sistem treba raditi i kako se podaci ponašaju.

Način izrade:

- Analizirati strukturu GTSRB dataset-a (train, test, meta).
- Pregledati raspodjelu klase i identifikovati nebalansiranost.
- Vizualizirati primjere slika radi razumijevanja boja, oblika i tipičnih problema.
- Identifikovati koje informacije iz meta podataka mogu pomoći (ShapeId, ColorId).

4.2.2. Korak 2 – Predobrada podataka

Ulagne slike potrebno je pripremiti za rad modela.

Planirana izrada:

- Standardizovati veličinu slika (32×32 , 48×48 ili 64×64).
- Normalizovati piksele (0–1 ili z-score).
- Primijeniti augmentaciju podataka: rotacije, zamjene, promjene osvjetljenja, horizontalni flip.
- Pripremiti tren/test validacijske setove.

4.2.3. Korak 3 – Segmentacija slike

Cilj je izdvojiti regije slike koje vjerovatno sadrže znak.

Način izrade:

Metoda A — HSV segmentacija boje

- Prebaciti sliku u HSV prostor.
- Definisati raspon crvene, plave i žute boje.
- Primijeniti maskiranje i morfološke operacije.
- Dobiti konture i izdvojiti moguće znakove.

Metoda B — Korištenje bounding boxa iz dataset-a

Metoda C — CNN za detekciju (YOLO ili SSD)

Moguća je i kombinacija više metoda.

4.2.4. Korak 4 – Ekstrakcija karakteristika

Cilj je pripremiti reprezentaciju znaka koju će model moći naučiti.

Planirana izrada:

- Klasične metode: HOG, LBP, boja, konture.
- Duboko učenje: CNN automatski uči karakteristike.

4.2.5. Korak 5 – Odabir modela i treniranje

Izbor modela koji model će klasifikovati znakove.

Način izrade:

- CNN (najbolja tačnost)
- LeNet5 varijante

- SVM + HOG
- ResNet/Fine-Tuning
- YOLO

Plan implementacije:

- Kreirati CNN model (3–4 konvolucijska bloka + dense slojevi).
- Koristiti CrossEntropyLoss i Adam optimizator.
- Trenirati kroz 10–50 epoha zavisno od vremena.
- Pratiti gubitak i tačnost na trening i validacijskom skupu.
- Po potrebi balansirati klase (oversampling ili class weights).

4.2.6. Korak 6 – Validacija i evaluacija modela

Procjenjuje se koliko dobro model radi na neviđenim podacima.

Planirana izrada:

- Evaluirati model na test skupu GTSRB.
- Prikazati metrike: Accuracy, Precision, Recall, Confusion Matrix.
- Analizirati greške — prikazati nekoliko pogrešno klasifikovanih znakova.
- Testirati model u različitim uslovima (rotacija, promjena boje, zamućenje).

4.2.7. Korak 7 – Implementacija sistema

Model se integriše u funkcionalni sistem koji prepoznaže znakove.

Planirana izrada:

- Implementirati skriptu za učitavanje slike i prikaz rezultata (OpenCV).
- Obezbijediti da se prepoznati znak i bounding box prikazuju na ekranu.
- Testirati brzinu obrade (FPS).

5. Predprocesiranje skupa podataka

5.1. Učitavanje skupa podataka

Jedan od prvih koraka bio je definisanje putanja. Kao osnovni direktorij projekta definisan je glavni direktorij (ROOT_DIR) koji predstavlja korijen radnog prostora na Google Drive-u. Ovaj direktorij služi kao centralna tačka za sve ostale direktorije i fajlove potrebne za rad, što omogućava bolju organizaciju i jednostavnije upravljanje dataset-om.

```
ROOT_DIR = '/content/drive/MyDrive/[POOS] - Projekat/'
DATASET_ROOT = os.path.join(ROOT_DIR, 'Dataset')
TRAIN_DIR = os.path.join(DATASET_ROOT, 'Train')
TEST_DIR = os.path.join(DATASET_ROOT, 'Test')
META_DIR = os.path.join(DATASET_ROOT, 'Meta')

META_CSV = os.path.join(DATASET_ROOT, 'Meta.csv')
TRAIN_CSV = os.path.join(DATASET_ROOT, 'Train.csv')
TEST_CSV = os.path.join(DATASET_ROOT, 'Test.csv')
```

Slika 7: Definisanje putanja

Proces predprocesiranja započinje učitavanjem ulaznog skupa podataka, koji obuhvata i tablične i slikovne podatke. Tablični podaci učitani su korištenjem biblioteke pandas, što omogućava efikasno upravljanje strukturiranim podacima i pregled njihove strukture kroz funkcije kao što su head(), info() i isna(). Paralelno s time, slikovni podaci učitani su

korištenjem biblioteka Pillow (PIL) i OpenCV, čime je omogućena naprednija vizuelna i tehnička obrada slika.

```

train_path = TRAIN_CSV if os.path.exists(TRAIN_CSV) else find_csv_file('Train.csv')
if train_path and os.path.exists(train_path):
    try:
        train_df = pd.read_csv(train_path)
        print(f"✓ Train.csv učitano sa: {train_path}")
        print(f"  Shape: {train_df.shape}")
    except Exception as e:
        print(f"✗ Greška pri učitavanju Train.csv: {e}")
else:
    print(f"✗ Train.csv nije pronađen")

test_path = TEST_CSV if os.path.exists(TEST_CSV) else find_csv_file('Test.csv')
if test_path and os.path.exists(test_path):
    try:
        test_df = pd.read_csv(test_path)
        print(f"✓ Test.csv učitano sa: {test_path}")
        print(f"  Shape: {test_df.shape}")
    except Exception as e:
        print(f"✗ Greška pri učitavanju Test.csv: {e}")
else:
    print(f"✗ Test.csv nije pronađen")

```

Slika 8: Učitavanje skupa podataka

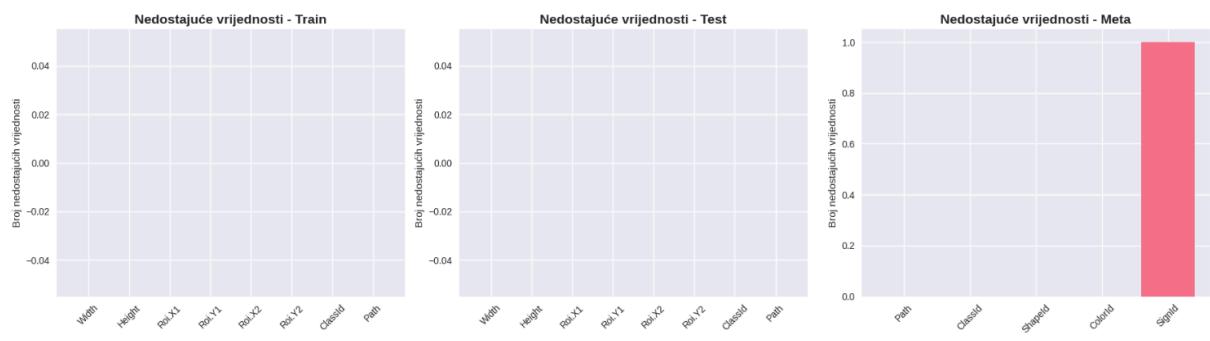
	Width	Height	Roi.X1	Roi.Y1	Roi.X2	Roi.Y2	ClassId	\
38357	41	45	5	6	35	39	39	
29817	96	93	8	9	88	85	25	
23860	33	39	6	5	28	34	15	
24901	26	31	5	6	21	26	17	
14543	63	63	6	6	58	58	9	
9234	47	41	5	6	41	35	5	
13547	39	40	6	6	34	35	8	
14602	52	51	5	5	47	46	9	
36678	42	44	5	5	36	38	38	
25574	29	27	5	6	24	22	18	
Path								
38357	Train/39/00039_00009_00018.png							
29817	Train/25/00025_00044_00027.png							
23860	Train/15/00015_00015_00010.png							
24901	Train/17/00017_00015_00001.png							
14543	Train/9/00009_00032_00023.png							
9234	Train/5/00005_00026_00024.png							
13547	Train/8/00008_00046_00017.png							
14602	Train/9/00009_00034_00022.png							
36678	Train/38/00038_00022_00019.png							
25574	Train/18/00018_00000_00014.png							

Slika 9: Prikaz slučajno izabranih podataka iz train skupa

Nakon što je uspješno učitan Train.csv sa informacijama o slikama za treniranje, na isti način su učitani i ostali podaci iz preostalih skupova.

5.2. Čišćenje podataka

Nakon učitavanja svih relevantnih CSV fajlova, izvršena je provjera nedostajućih vrijednosti u svim skupovima podataka. Za svaki od Train, Test i Meta skupova podataka provjerava se da li DataFrame postoji, a zatim se računa ukupan broj nedostajućih vrijednosti po koloni.



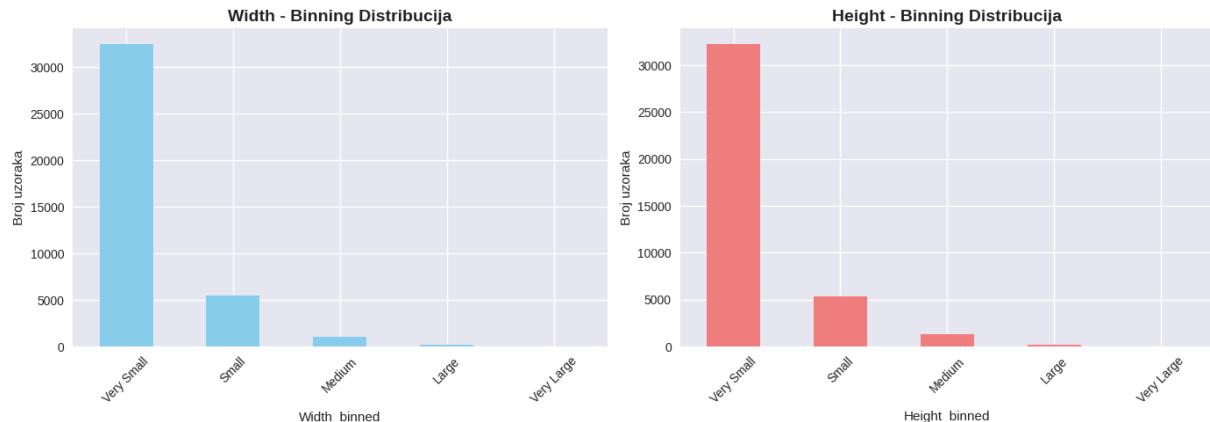
Slika 10: Histogram nedostajućih vrijednosti

Ako neka kolona sadrži nedostajuće vrijednosti (NaN), te vrijednosti se popunjavaju srednjom vrijednošću te kolone.

Binning

Binning je proces pretvaranja kontinuiranih numeričkih vrijednosti u diskrete kategorije ili intervale. Umjesto da se radi sa svakom pojedinačnom vrijednošću, vrijednosti se grupišu u "binove" ili intervale, čime se pojednostavljuje analiza podataka i smanjuje utjecaj ekstremnih vrijednosti.

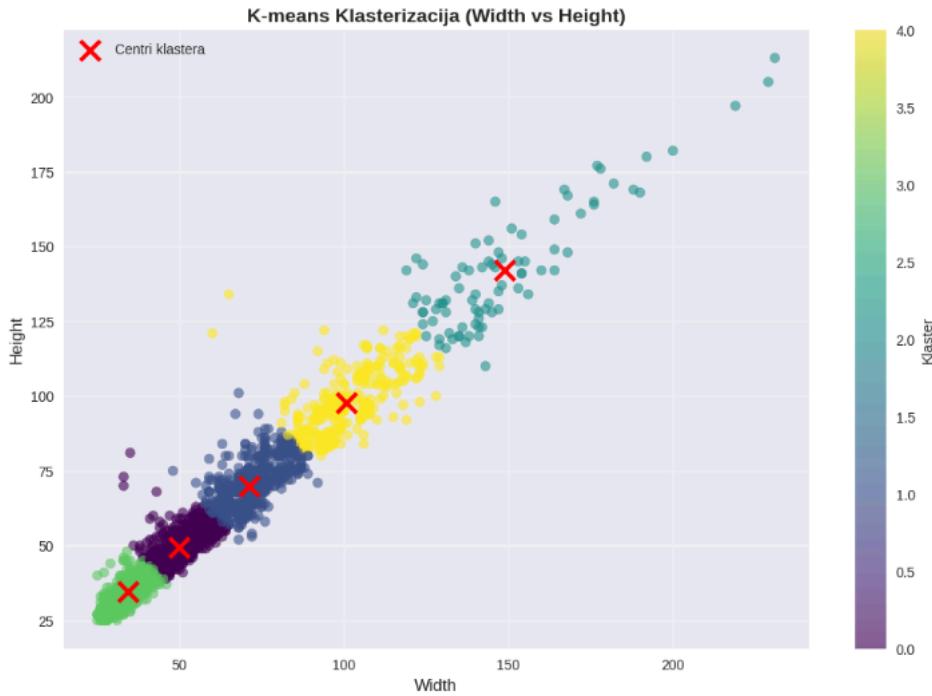
Vrijednosti u tim kolonama se dijele u pet intervala (binova), a svakom intervalu se dodjeljuje kategorijalna oznaka od Very Small do Very Large.



Slika 11: Binning

Klasterizacija

Klasterizacija je izvšena koristeći algoritam K-means kako bi grupisao uzorke na osnovu dimenzija Width i Height. Prvo se provjerava da li te kolone postoje u skupu podataka, zatim se uzima nasumični uzorak do 5000 redova radi brže analize. Zatim se primjenjuje K-means sa pet klastera. Svaki uzorak se dodjeljuje jednom od klastera, a centri klastera se također izračunavaju.



Slika 12: Klasterizacija

5.3. Transformacija podataka

Normalizacija

Izvršena je normalizacija i standardizacija podataka, na način da se kreira kopija skupa podataka kako bi se sačuvala originalna verzija. Zatim se primjenjuje Min-Max normalizacija koja skalira vrijednosti odabrane numeričke kolone tako da budu unutar intervala od 0 do 1. To omogućava da sve numeričke varijable budu usporedive i da se spriječi dominacija kolona sa većim vrijednostima u algoritmima koji zavise od udaljenosti ili sličnosti. Nakon toga, na istim kolonama se primjenjuje standardizacija (Z-score), koja transformira vrijednosti tako da imaju srednju vrijednost 0 i standardnu devijaciju 1

```
Min-Max normalizacija (0-1) primijenjena na:
Width: min=0.0000, max=1.0000
Height: min=0.0000, max=1.0000
Roi.X1: min=0.0000, max=1.0000
Roi.Y1: min=0.0000, max=1.0000
Roi.X2: min=0.0000, max=1.0000
Roi.Y2: min=0.0000, max=1.0000

Standardizacija (z-score) primijenjena
Width: mean=0.0000, std=1.0000
Height: mean=0.0000, std=1.0000
Roi.X1: mean=0.0000, std=1.0000
Roi.Y1: mean=-0.0000, std=1.0000
Roi.X2: mean=0.0000, std=1.0000
Roi.Y2: mean=-0.0000, std=1.0000
```

Slika 13: Normalizacija i standardizacija

5.4. Smanjenje skupa podataka

Početni trening skup sadrži ukupno 39 209 zapisa, raspoređenih u veći broj klasa s neravnomjernom distribucijom uzorka. Zbog velike količine podataka, kao i potrebe za

smanjenjem računarskog opterećenja tokom treniranja modela, izvršena je redukcija skupa podataka. Redukcija je provedena tako što je izabrano po 20 reprezentativnih uzoraka iz svake klase, čime je osigurana ujednačena i balansirana zastupljenost klasa u trening skupu. Ovakav pristup omogućava brže treniranje modela, lakšu analizu rezultata i smanjuje rizik od pristrasnosti prema klasama s većim brojem uzoraka, uz zadržavanje osnovnih karakteristika originalnog skupa podataka.

6. Model neuronske mreže

6.1. Odabir arhitekture

U projektu koristimo CNN arhitekturu zasnovanu na LeNet modelu. Ova arhitektura je pogodna za rad sa slikama male rezolucije, efikasna je za treniranje i dovoljno je jednostavna da se primjeni na ograničen broj primjera u trening skupu.

Ulazne slike se skaliraju na veličinu 45x45x3, kako bi dimenzije bile uniformne kroz cijeli dataset.

6.2. Struktura neuronske mreže

Mreža se sastoji od sljedećih slojeva:

- **Ulazni sloj:** 45x45x3
- **1. konvolucijski sloj:**
 - 32 filtera 3x3, stride 1, bez paddinga
 - Batch Normalizationi sloj
 - Izlaz: 43x43x32
- **1. max-pooling sloj:**
 - Jezgra 2x2
 - Izlaz: 21x21x32
- **2. konvolucijski sloj:**
 - 64 filtera 5x5, stride 1, bez paddinga
 - Izlaz: 17x17x64
- **2. max-pooling sloj:**
 - Jezgra 2x2
 - Izlaz: 8x8x64
- **Potpuno povezani slojevi:**
 - Flatten sloj: 4096
 - FC1: 4096 → 256 (uz Dropout 0.3)
 - FC2: 256 → 128 (uz Dropout 0.3)
- **Izlazni sloj:**
 - 128 → 43 (jedan izlaz po klasi)

6.3. Aktivacijske funkcije i inicijalizacija

- Aktivacija svih slojeva osim izlaznog: ReLU
- Aktivacija izlaza: Softmax
- Inicijalizacija težina: uniformna distribucija

6.4. Funkcija gubitka i optimizator

- Funkcija gubitka: Sparse Categorical Crossentropy
- Optimizator: Adam, početna stopa učenja 0.001
- Planirano treniranje: 35 epoha, uz mogućnost povećanja ako performanse budu nezadovoljavajuće

7. Rad sa skupom podataka

7.1. Učitavanje skupa podataka

Dataset se sastoji od:

- trening skupa
- testnog skupa
- metapodataka o klasama

Sve slike su učitane i zatim skalirane na dimenzije 45x45x3.

7.2. Reduciranje trening skupa

Zbog velikog broja slika, trening skup je ograničen na:

- 100 slika po klasi
- Ukupno 4300 slika za treniranje (100x43)

7.3. Podjela na trening, validacijski i testni skup

Trening i validacijski skup izdvajaju se iz reduciranog trening dijela GTSRB skupa, dok se za finalno testiranje koristi orginalni testni skup:

- Trening skup: 3440 slika
- Validacijski skup: 860 slika

Podaci su dobro uravnoteženi, što je pogodno za metrike koje koristimo.

8. Evaluacija modela

8.1. Metrike performansi

Za evaluaciju modela koristimo:

- tačnost (accuracy)
- preciznost (precision)
- odziv (recall)
- F1-score

Ove metrike daju jasan uvid u kvalitet klasifikacije za sve klase.

Klasa	Precision	Recall	f1-score	Support
0	0.88	0.83	0.85	60.00
1	0.98	0.88	0.93	720.00
2	0.91	0.95	0.93	750.00
3	0.80	0.92	0.85	450.00
4	0.94	0.97	0.96	660.00
5	0.77	0.88	0.82	630.00
6	0.98	0.83	0.90	150.00
7	0.92	0.79	0.85	450.00
8	0.88	0.79	0.83	450.00
9	0.94	0.99	0.97	480.00
10	0.99	0.91	0.95	660.00
11	0.97	0.85	0.90	420.00
12	0.98	0.91	0.94	690.00
13	0.99	0.99	0.99	720.00
14	1.00	0.99	0.99	270.00
15	0.79	1.00	0.88	210.00
16	0.94	1.00	0.97	150.00
17	0.99	0.97	0.98	360.00
18	0.95	0.91	0.93	390.00
19	0.83	1.00	0.91	60.00
20	0.64	0.99	0.78	90.00
21	0.87	0.80	0.83	90.00
22	0.98	0.88	0.93	120.00
23	0.84	0.96	0.90	150.00
24	0.96	0.98	0.97	90.00

25	0.96	0.95	0.95	480.00
26	0.92	0.83	0.87	180.00
27	0.90	0.93	0.92	60.00
28	0.90	0.97	0.93	150.00
29	0.73	1.00	0.84	90.00
30	0.76	0.65	0.70	150.00
31	0.91	0.96	0.93	270.00
32	0.65	1.00	0.79	60.00
33	1.00	1.00	1.00	210.00
34	1.00	0.99	1.00	120.00
35	0.99	0.98	0.99	390.00
36	0.99	0.86	0.92	120.00
37	0.98	0.98	0.98	60.00
38	0.96	0.97	0.96	690.00
39	1.00	1.00	1.00	90.00
40	0.97	0.94	0.96	90.00
41	0.94	0.78	0.85	60.00
42	0.98	0.93	0.95	90.00
accuracy			0.92	12630
macro avg	0.91	0.92	0.91	12630
weighted avg	0.93	0.92	0.92	12630

Tabela 10: Evaluacija modela

8.1.1. Eksperimentisanje sa veličinom slike

Prilikom testiranja sa različitim veličinama slika, dobijamo sljedeće rezultate:

Parametri:

- 35 max epoha
- Patience za early stop = 8

za slike dimenzija 30x30

- Train accuracy: 98.2%
- Validation accuracy: 95.1%
- Test accuracy: 90.4%
- Gap: 3.1%
- 24 Epoha(early stop)

za 45x45

- Train accuracy: 98.37%
- Validation accuracy: 96.40%
- Test accuracy: 91.94%
- Gap: 2%
- 20 Epoha(early stop)

za 60x60

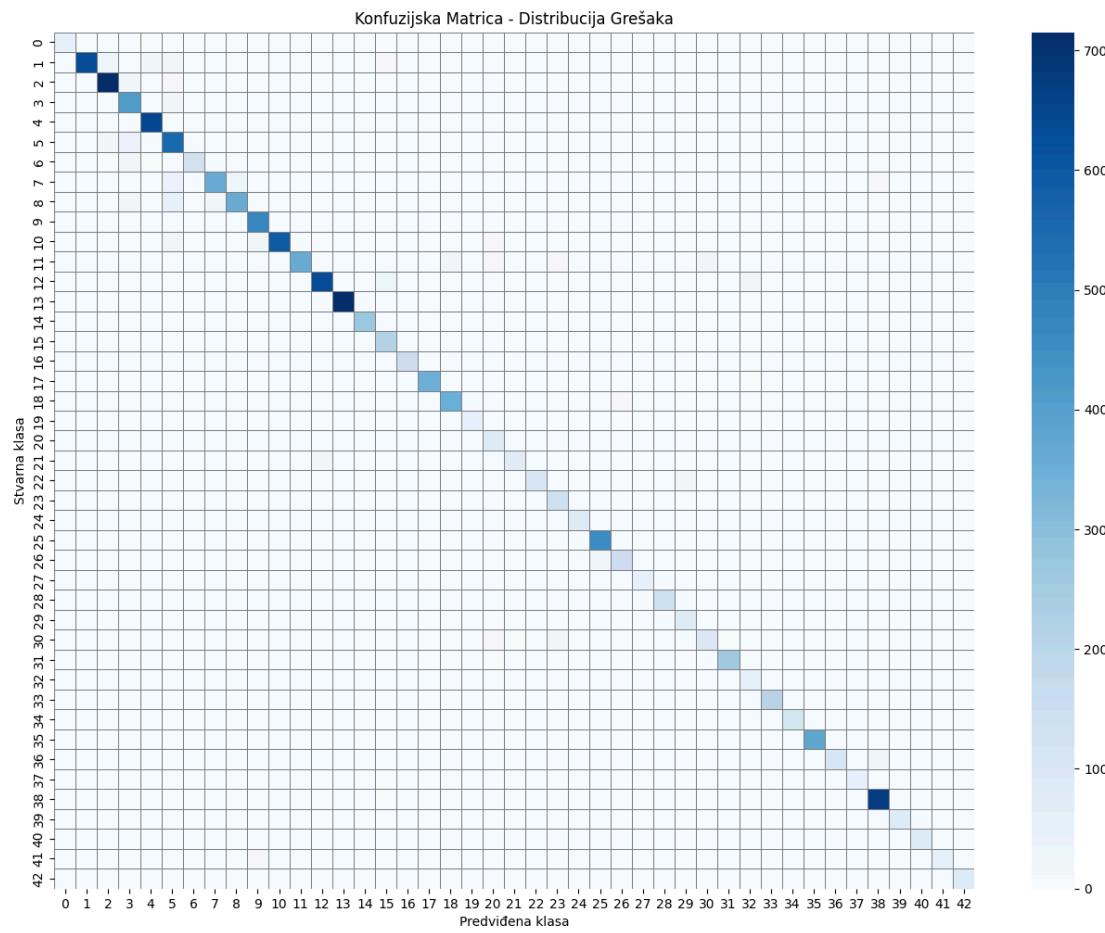
- Train accuracy: 99.1%
- Validation accuracy: 97.1%
- Test accuracy: 90.87
- Gap: 2.2%
- 23 Epoha(early stop)

Primjetno je znatno usporenje treniranja modela sa povećanjem veličine slika, a pritom nemamo neke značajnije benefite.

8.2. Konfuzijska matrica

Pored osnovnih metrika, koristićemo konfuzijsku matricu, kako bismo utvrdili:

- koje klase model najčešće zbujuje
- gdje se pojavljuju najveće greške
- kako se performanse razlikuju između klasa



Slika 14: Konfuzijska matrica

8.3. Vizualni prikaz predikcija

Prikazujemo 10 nasumičnih predikcija i vidimo da je 8 tačnih, a u dvije predikcije su napravljene greške.



Slika 15: Vizualni prikaz predikcija

9. Implementacija grafičkog korisničkog interfejsa

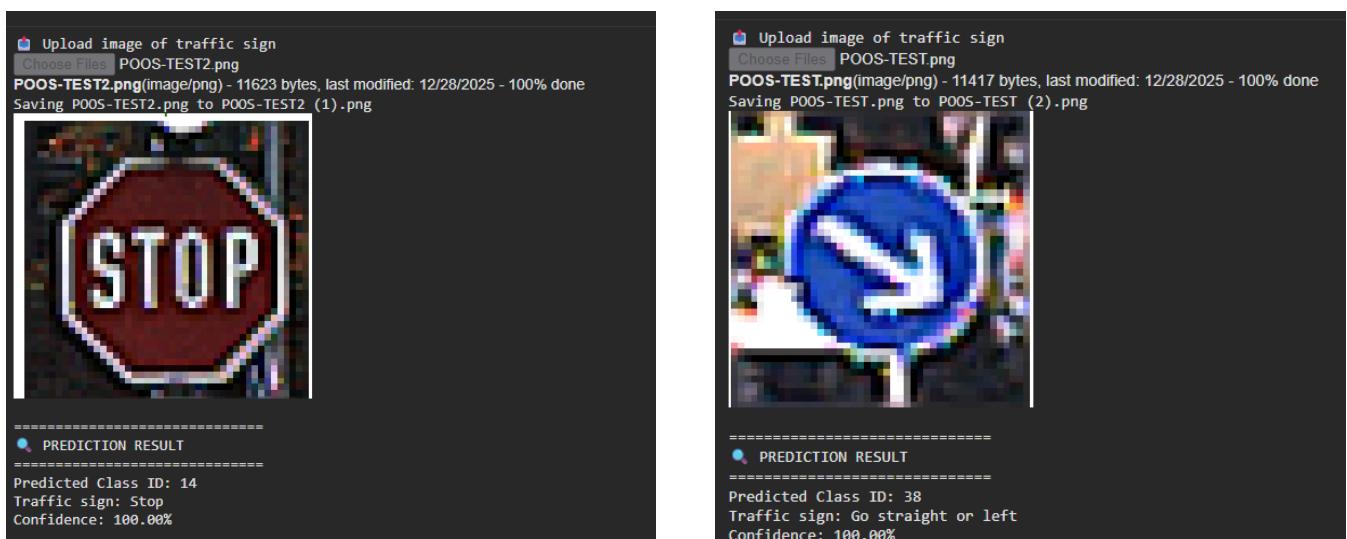
9.1 Opis zadatka

Cilj projekta je izrada UI za prepoznavanje saobraćajnih znakova pomoću treniranog modela dubokog učenja.

UI uraden preko dva nacina:

- putem Google Colab okruženja
- putem GUI (Visual Studio Code)

9.2 Način 1: Google Colab



Slika 16: UI preko Google Colab

9.2.1 Korištene biblioteke

U radu su korištene sljedeće Python biblioteke:

- **Google Colab (files)** - za učitavanje slike sa računara
- **IPython.display** - za prikaz učitane slike
- **PIL (Python Imaging Library)** - za obradu slike
- **NumPy** - za rad s numeričkim podacima
- **TensorFlow** - za predikciju pomoću neuronske mreže

9.2.2 Predobrada slike

Definisana je funkcija preprocess_image(img) koja vrši sljedeće korake:

1. **Konverzija u RGB format** - osigurava da slika ima tri kanala boje
2. **Poboljšanje ivica (EDGE_ENHANCE)** - naglašava oblike znaka
3. **Promjena dimenzija na 45×45 piksela** - prilagođavanje ulazu modela
4. **Normalizacija vrijednosti piksela** - dijeljenjem sa 255 vrijednosti se svode u opseg 0–1
5. **Dodavanje dimenzije (batch size)** - potrebno za rad s modelom

9.2.3 Učitavanje slike

Korisnik učitava sliku saobraćajnog znaka pomoću: files.upload()

Nakon učitavanja, slika se:

- otvara pomoću PIL.Image
- prikazuje na ekranu radi provjere ispravnosti

9.2.4 Predikcija pomoću modela

Nakon predobrade slike:

- Slika se prosljeđuje u **istrenirani model**
- Model vraća vjerovatnoće za sve klase
- Određuje se:
 - **Predviđena klasa** (argmax)
 - **Stepen sigurnosti (confidence)**

Rezultat se ispisuje u čitljivom formatu.

9.2.5 Izlaz programa

Program ispisuje:

- **ID predviđene klase saobraćajnog znaka**
- **Naziv znaka**
- **Procenat sigurnosti predikcije**

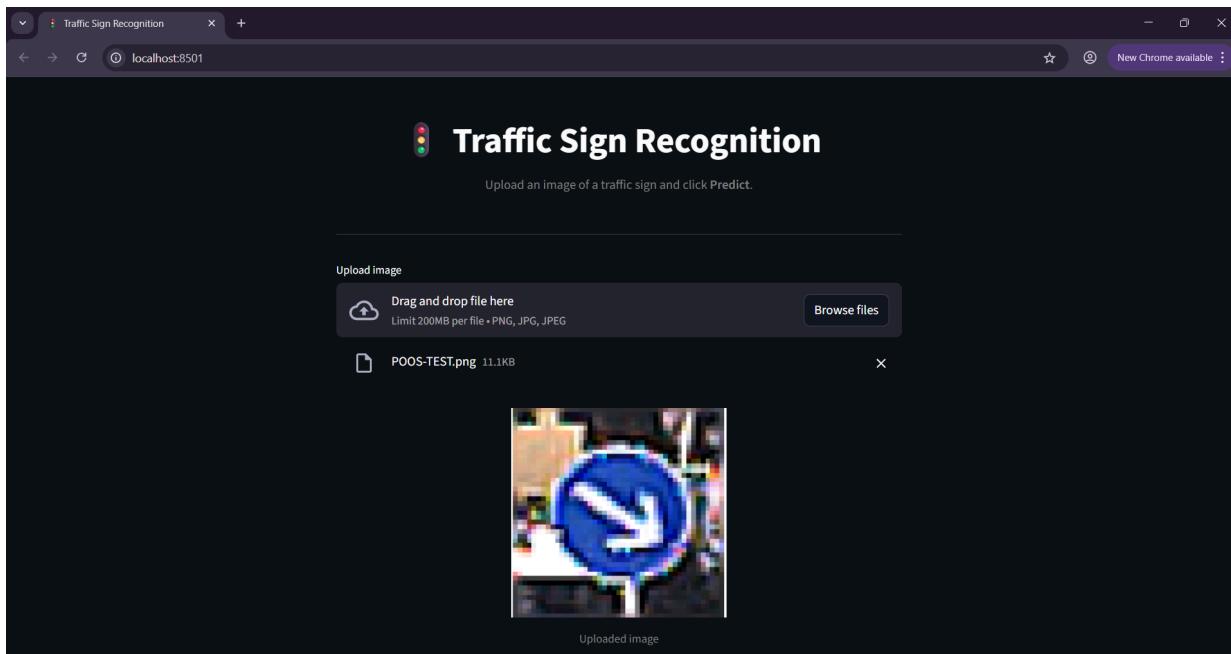
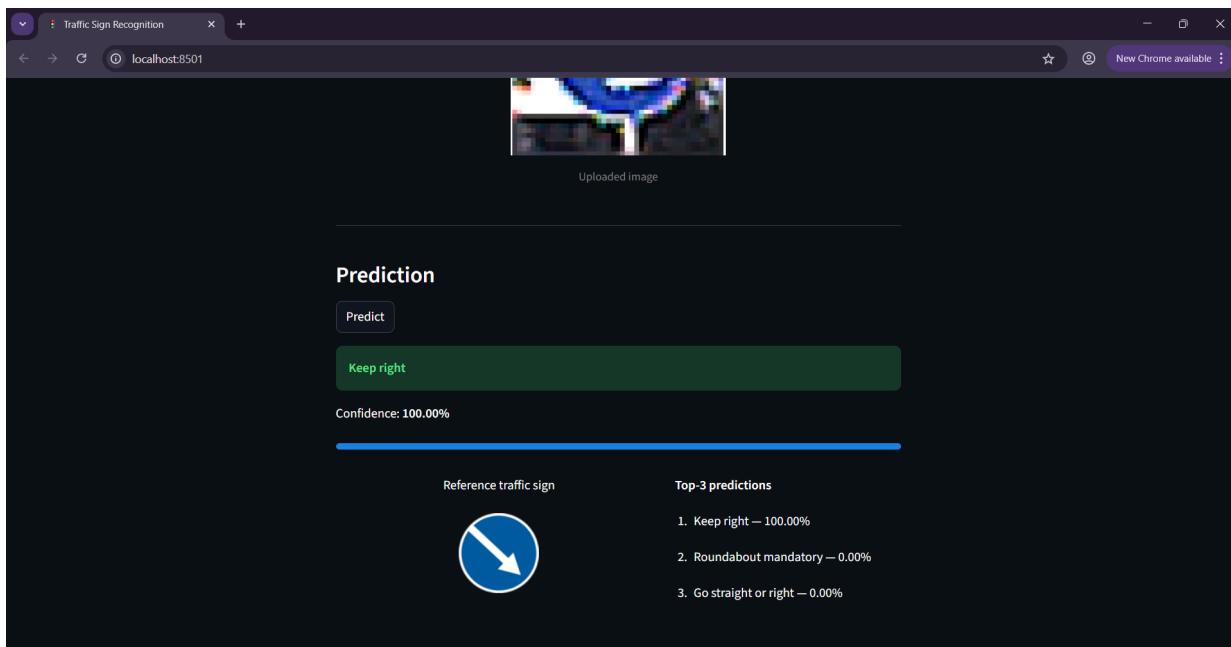
9.2.6 Zaključak

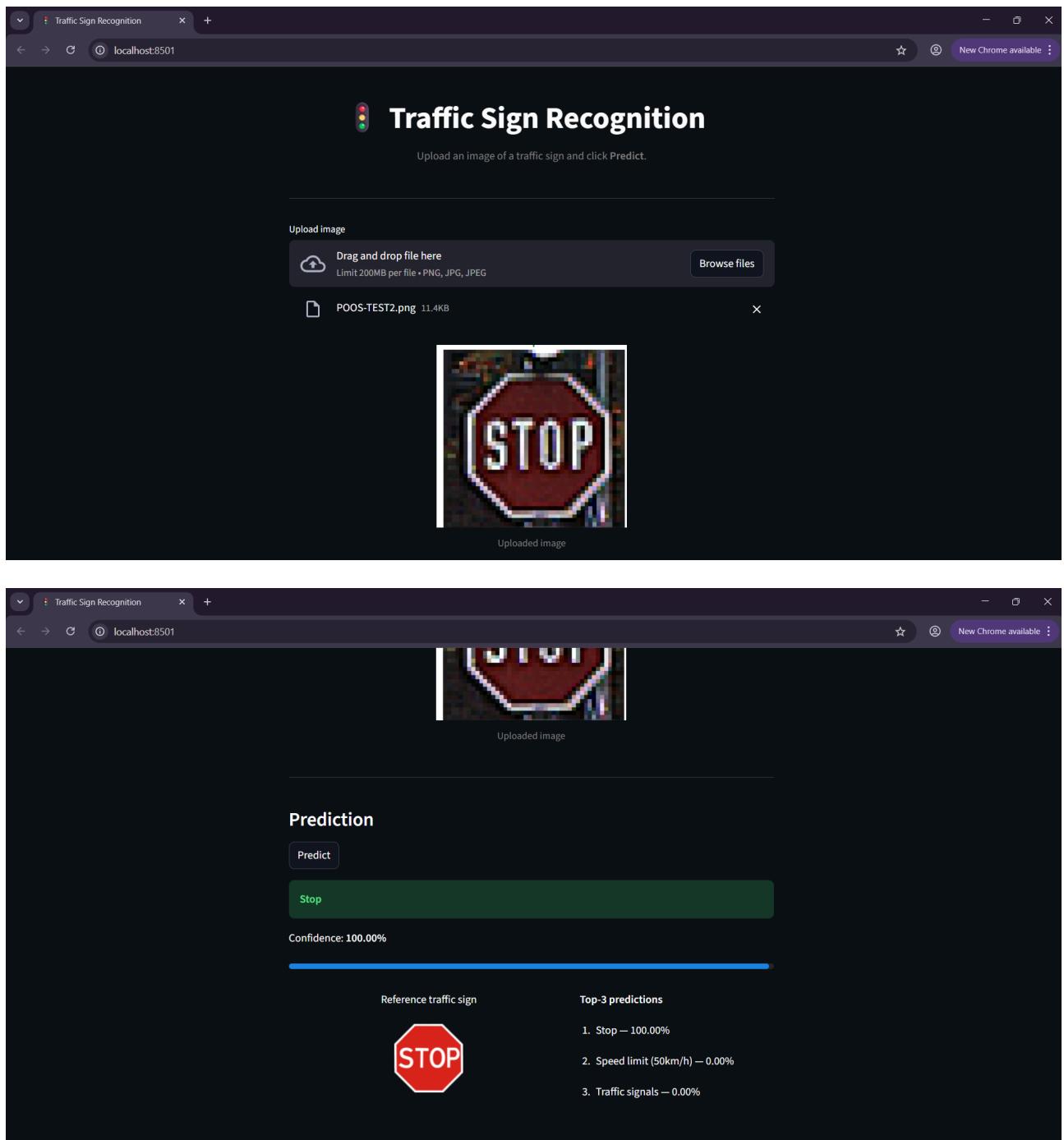
Ovim zadatkom uspješno je demonstriran **kompletan proces klasifikacije slike**:

- od učitavanja,
- preko obrade,
- do predikcije pomoću neuronske mreže.

Sistem se može koristiti kao osnova za **automatsko prepoznavanje saobraćajnih znakova**, što je važan dio inteligentnih transportnih sistema i autonomne vožnje.

9.3 Način 2: Visual Studio Code – GUI aplikacija





Slika 17: UI preko GUI

9.3.1 Cilj GUI aplikacije

Cilj GUI dijela je omogućiti korisniku jednostavno korištenje modela:

- upload slike,
- prikaz predikcije + naziv,
- prikaz confidence vrijednosti,
- prikaz Top-3 predikcija,
- prikaz referentne slike znaka iz dataset-a (Meta folder).

9.3.2 Korištene biblioteke

- **Streamlit (streamlit)** – izrada web aplikacije i korisničkog interfejsa
- **TensorFlow (tensorflow)** – učitavanje i korištenje istreniranog modela
- **NumPy (numpy)** – rad sa nizovima i vjerovatnoćama predikcije
- **PIL (Pillow)** – otvaranje i obrada slika
- **os / pathlib** – provjera i pronađak fajlova (model + Meta slike)
- **base64 + BytesIO** – prikaz slika centrirano u Streamlit-u pomoću HTML-a

9.3.3 Podešavanje stranice (UI)

Urađeno je:

- postavljanje naslovne ikone, naziva i layout-a stranice pomoću `st.set_page_config`
- dodan centrirani naslov i kratak opis aplikacije.

9.3.4 Model i konfiguracija

Definisano je:

- putanja do modela: `traffic_sign_model.h5`
- putanja do foldera referentnih slika: `Meta`
- veličina ulazne slike: **45×45**

Aplikacija prvo provjerava da li model postoji:

- ako ne postoji → ispisuje grešku i prekida rad (`st.stop()`)

Model se učitava funkcijom `load_model()` koja je keširana sa `@st.cache_resource`, što znači da se model ne učitava iznova pri svakom osvježavanju aplikacije (brže i efikasnije).

9.3.5 Mapa klasa (GTSRB)

U kodu postoji rječnik `CLASSES` koji mapira:

- **ID klase (0–42) → naziv znaka (tekst)**

Funkcija `label_name()` vraća naziv znaka za dati ID.

9.3.6 Predobrada slike

Funkcija `preprocess(img)` radi isto kao u treningu:

1. pretvara sliku u RGB
2. pojačava ivice (EDGE_ENHANCE)
3. resize na **45×45**
4. normalizacija (0–1)
5. dodaje batch dimenziju (oblik pogodan za model)

9.3.7 Upload i prikaz slike

Korisnik učitava sliku preko:

- st.file_uploader(...)

Nakon toga slika se:

- otvara Image.open(uploaded)
- prikazuje centrirano (HTML + base64) bez “zoom” opcija, uz fiksnu širinu (300px)

9.3.8 Predikcija i prikaz rezultata

Kada korisnik klikne **Predict**:

- prikazuje se spinner “Running prediction...”
- radi se predikcija:
 - probs = model.predict(x)[0]
- uzima se:
 - pred_id = klasa sa najvećom vjerovatnoćom (argmax)
 - confidence = procenat sigurnosti

Rezultat se prikazuje kao:

- st.success (naziv znaka)
- tekst sa procentom sigurnosti
- progress bar (st.progress)

9.3.9 Referentna slika (Meta folder)

Funkcija find_meta_image_for_class(class_id) pokušava pronaći primjer slike znaka:

- traži fajl tipa Meta/14.png (ili jpg/jpeg)
- ili traži sliku u folderu Meta/14/

Ako pronađe → prikaže referentnu sliku (100px) pored rezultata.

Ako ne pronađe → ispiše “No example image available.”

9.3.10 Top-3 predikcije

Aplikacija dodatno ispisuje **tri najvjerojatnije klase**:

- sortira vjerovatnoće
- uzima top 3
- ispisuje naziv i procenat za svaku

9.3.11 Zaključak

U ovom zadatku napravili smo funkcionalnu Streamlit web aplikaciju koja omogućava prepoznavanje saobraćajnih znakova koristeći istrenirani TensorFlow model. Aplikacija je user-friendly jer prikazuje:

- učitanu sliku,
- glavnu predikciju + confidence,
- top-3 opcije,
- referentni primjer znaka (Meta).

9.4 Razlika između Colab i GUI

Prvi način (Google Colab) služi za **testiranje modela** i koristi se kroz notebook bez grafičkog interfejsa, gdje se rezultati ispisuju tekstualno.

Drugi način (Streamlit) je **web aplikacija** sa grafičkim interfejsom, dugmetom za predikciju i dodatnim prikazima (confidence, top-3, referentna slika), te je namijenjen krajnjim korisnicima.

Lista slika

Slika 1: Prikaz saobraćajnih znakova	5
Slika 2: Primjeri slika iz GTSRB skupa podataka koji sadrži 43 klase saobraćajnih znakova.	12
Slika 3: Struktura YOLOv5 modela i način prenosa informacija između slojeva.	13
Slika 4: Konverzija RGB slike u HSV sliku	16
Slika 5: Poboljšana verzija LuNet-5 modela	17
Slika 6: Grupacija saobraćajnih znakova	17
Slika 7: Definisanje putanja	27
Slika 8: Učitavanje skupa podataka	28
Slika 9: Prikaz slučajno izabralih podataka iz train skupa	28
Slika 10: Histogram nedostajućih vrijednosti	29
Slika 11: Binning	29
Slika 12: Klasterizacija	30
Slika 13: Normalizacija i standardizacija	30
Slika 14: Konfuzijska matrica	36
Slika 15: Vizualni prikaz predikcija	37
Slika 16: UI preko Colab	39
Slika 17: UI preko GUI	42

Lista tabela

Tabela 1: Opis i prikaz korištenih metoda	4
Tabela 2: Prikaz rezultata korištenih metoda	10
Tabela 3: Prikaz rezultata klasifikacije	18
Tabela 4: Broj slika po klasama u trening skupu	20
Tabela 5: Broj slika po klasama u test skupu	21
Tabela 6: Trening skup (Train.csv) i test skup (Test.csv)	21
Tabela 7: Meta podaci (Meta.csv)	22
Tabela 8: Osnovne statistike za numeričke kolone	22
Tabela 9: Fizičke karakteristike slika	23
Tabela 10: Evaluacija modela	34