Universitatea POLITEHNICA din București

Facultatea de Electronică, Telecomunicații și Tehnologia Informației

Aplicație de detecție și identificare a semnelor de circulație

Proiect de Diplomă

Prezentat ca cerință parțială pentru obținerea

titlului de *Inginer*

în domeniul *Electronică și Telecomunicații*

programul de studii *Tehnologii și Sisteme de Telecomunicații*

Conducător științific

Conf.Dr.Ing. Ionuţ PIRNOG

Absolvent

Popescu Ervin-Adrian

Anul 2022

Universitatea “Politehnica” din Bucureşti **Anexa 1**
Facultatea de Electronică, Telecomunicaţii şi Tehnologia Informaţiei
Program de studiu **TST**

TEMA PROIECTULUI DE DIPLOMĂ
a studentului **POPESCU A. Ervin-Adrian, 444C**

1. Titlul temei: Aplicație de detecție și identificare a semnelor de circulație

2. Descrierea temei și a contribuției personale a studentului (în afara părții de documentare):
Se va implementa o aplicație de detecție și identificare a semnelor de circulație în imagini și secvențe video. Aplicația se poate implementa în Matlab, C , Python, Java. Se pot folosi librării specifice și algoritmi dedicați prelucrării imaginilor/video: OpenCV, YOLOv4, Pytorch, Tensorflow, Python Tesseract, etc..

3. Discipline necesare pt. proiect:
PDS; POO; TCSM

4. Data înregistrării temei: 2023-02-03 18:43:47

Conducător(i) lucrare,
Conf.Dr.Ing. Ionuţ PIRNOG

Student,
POPESCU A. Ervin-Adrian

Director departament,
Conf. dr. ing. Șerban OBREJA

Decan,
Prof. dr. ing. Mihnea UDREA

Cod Validare: **6434b356e1**

Declaraţie de onestitate academică

Prin prezenta declare că lucrarea cu titlul *Aplicație de detecție și identificare a semnelor de circulație*, prezentată în cadrul Facultăţii de Electronică, Telecomunicaţii şi Tehnologia Informaţiei a Universităţii "Politehnica" din Bucureşti ca cerinţă parţială pentru obţinerea titlului de *Inginer* în domeniul Inginerie Electronică şi Telecomunicaţii/ Calculatoare şi Tehnologia Informaţiei, programul de studii *Tehnologii și Sisteme de Telecomunicații* este scrisă de mine şi nu a mai fost prezentată niciodată la o facultate sau instituţie de învăţământ superior din ţară sau străinătate. Declare că toate sursele utilizate, inclusiv cele de pe Internet, sunt indicate în lucrare, ca referinţe bibliografice. Fragmentele de text din alte surse, reproduse exact, chiar şi în traducere proprie din altă limbă, sunt scrise între ghilimele şi fac referinţă la sursă. Reformularea în cuvinte proprii a textelor scrise de către alţi autori face referinţă la sursă. Înţeleg că plagiatul constituie infracţiune şi se sancţionează conform legilor în vigoare. Declare că toate rezultatele simulărilor, experimentelor şi măsurătorilor pe care le prezint ca fiind făcute de mine, precum şi metodele prin care au fost obţinute, sunt reale şi provin din respectivele simulări, experimente şi măsurători. Înţeleg că falsificarea datelor şi rezultatelor constituie fraudă şi se sancţionează conform regulamentelor în vigoare.

Bucureşti, Iulie 2022.

Absolvent: Popescu Ervin-Adrian

.....

Cuprins

Lista figurilor

Lista tabelelor

Lista acronimelor

CNN: Convolutional Neural Network

Capitolul 1

Introducere

Semnele de circulație joacă un rol vital în menținerea siguranței rutiere și a fluidității traficului. Detectarea și identificarea acestor semne poate fi o sarcină dificilă și de multe ori costisitoare, deoarece necesită o analiză vizuală atentă a imaginilor și secvențelor video. În această lucrare de diploma, se va propune o aplicație de detecție și identificare a semnelor de circulație utilizând diverse librării și algoritmi dedicați prelucrării imaginilor/video, cum ar fi OpenCV, YOLOv4, Pytorch, TensorFlow și Python Tesseract. Această aplicație va permite o detectare precisă și rapidă a semnelor de circulație, îmbunătățind astfel siguranța rutieră și eficiența traficului.

Contribuția personală a acestui proiect constă în implementarea și optimizarea unui sistem de recunoaștere a semnelor de circulație în imagini și secvențe video. Proiectul va fi implementat în Python, utilizând diferite librării și algoritmi specifici de prelucrare a imaginilor și algoritmici de machine learning. Acest sistem va fi capabil să detecteze și să identifice semnele de circulație cu o precizie ridicată, prin utilizarea unor modele de învățare profundă, cum ar fi rețele neuronale convoluționale (CNN). În plus, aplicația va fi optimizată pentru a asigura o viteză de procesare ridicată, ceea ce va permite utilizarea sa în timp real în diferite situații de trafic.

În concluzie, această lucrare de diplomă va prezenta o aplicație inovatoare de detecție și identificare a semnelor de circulație, care va îmbunătăți siguranța rutieră și eficiența traficului. Prin implementarea și optimizarea unui sistem de recunoaștere a semnelor de circulație în imagini și secvențe video, acest proiect va reprezenta o contribuție semnificativă la domeniul prelucrării imaginilor și al recunoașterii de modele.

Anexa A

Cod sursă

```
import gzip
import json
import math
import os
import pathlib
import pickle
import random
import subprocess

import cv2
import ffmpeg
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
from keras.applications import Xception
from keras.layers import Dense, Dropout, Flatten, Input
from keras.models import Model, load_model
from keras.optimizers.rmsprop import RMSprop
from keras.utils import img_to_array, load_img
from keras.utils.vis_utils import plot_model
```

A.1: Import-uri

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelBinarizer

current_file_path = pathlib.Path(__file__).parent
input_path = os.path.join(current_file_path, "input")
output_path = os.path.join(current_file_path, "output")


# Define the location of the dataset
training_data_dir = os.path.join(input_path, "images", "Training")
test_data_dir = os.path.join(input_path, "images", "Test")
input_videos_dir = os.path.join(input_path, "videos")

```

```
# Define the image size and number of classes
IMG_SIZE = (64, 64)
VIDEO_SIZE = (1024, 1024)
NUM_CLASSES = 43
INIT_LR = 1e-3
```

A.2: Definiții de constante si path-uri

```
BATCH_SIZE = 64
labels_path = os.path.join(input_path, "labels.json")
input_videos_filenames = os.listdir(input_videos_dir)
lb_path = os.path.join(output_path, "lb.pickle")
predicted_labels_path = os.path.join(output_path, "predicted_labels.pickle")
saved_model_path = os.path.join(output_path, "model.h5")
scores_path = os.path.join(output_path, "scores.txt")


class VideoWriter:
    def __init__(
        self,
        fn,
        vcodec="libx264",
        fps=60,
        in_pix_fmt="rgb24",
        out_pix_fmt="yuv420p",
        input_args=None,
        output_args=None,
    ):
        self.fn = fn
        self.process: subprocess.Popen = None
        self.input_args = {} if input_args is None else input_args
        self.output_args = {} if output_args is None else output_args
        self.input_args["framerate"] = fps
        self.input_args["pix_fmt"] = in_pix_fmt
        self.output_args["pix_fmt"] = out_pix_fmt
        self.output_args["vcodec"] = vcodec

    def add(self, frame: np.ndarray):
        if self.process is None:
            h, w = frame.shape[:2]
            self.process = (
                ffmpeg.input(
                    "pipe:",
                    format="rawvideo",
                    s="{}x{}".format(w, h),
                    **self.input_args,
                )
```

```
40                 .output(self.fn, **self.output_args)
41                 .overwrite_output()
42                 .run_async(pipe_stdin=True)
43             )
44             # print(self.process.args)
```

A.3: Funcția care încarcă datele