BỘ THÔNG TIN VÀ TRUYỀN THÔNG

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

5th Report

Foundation Internship

Project Title: Traffic License Recognition

Instructor: Kim Ngoc Bach

Student Name : Bui Xuan Hai

Student ID : B22DCAT105

Lớp : E22CQCN05-B

Hà Nội - 2025

INTERNSHIP BASE REPORT

I. Project Introduction

This project aims to develop a web application that supports automatic recognition of vehicle license plates from uploaded images or videos. The system uses a deep learning model (trained by myself) to detect and recognize license plate numbers. This can be applied in real-life scenarios such as entry/exit management in parking lots or garages.

Users can interact with the system via a web interface, upload photos or videos, and the system will display the detected license plates. The backend is implemented using Flask, while the frontend is designed using HTML, CSS, and JavaScript (with optional integration of Bootstrap or Tailwind CSS).

II. Key Features

- Allow users to upload vehicle images or videos.
- Automatically detect and recognize license plate numbers.
- Display recognition results in a user-friendly format.
- Possibility to store recognition history or export results.
- Designed for application in garage or parking management systems.

III. Technologies Used

- **Frontend**: HTML, CSS, JavaScript (Bootstrap/Tailwind CSS).
- **Backend**: FastAPI (Python).
- **AI Model**: YOLO-based license plate detection + OCR model (e.g., PaddleOCR or custom model).
- **Storage**: Local server or database (SQLite/MySQL)

IV. Week 5

1. **Weekly goals**

- Build backend API using FastAPI
- Create endpoints to receive images and return license plate recognition results
- Connect YOLOv5 and PaddleOCR models to the processing flow from API

2. **Work done**

- Install and initialize FastAPI project
- Install necessary libraries: fastapi, uvicorn, paddleocr, torch, opencv-python, numpy
- Configure CORS to allow frontend to access API
- Create main endpoints
- Each endpoint supports POST with image files sent from frontend or test tools like Postman
- Integrate AI pipeline into FastAPI
- Connect YOLOv5 (using torch.load or torch.hub.load) to detect license plate areas
- Use PaddleOCR to recognize strings from detected areas
- Process output and respond to results
- API returns JSON including: status, recognized_text, bounding_box, confidence
- Add exception handling (try-except) to ensure stability when the image is faulty or not detected

3. **Results achieved**

- The FastAPI backend system has been operating stably
- Users can upload images and receive back a string of recognized license plates
- Communication between parts in the pipeline is smooth
- Can be tested directly via Swagger UI or Postman

4. **Difficulties encountered**

- The initialization process of the YOLOv5 model has a delay (a few seconds) → needs to be preloaded at startup
- Processing input images from UploadFile needs to be converted accurately via PIL → np.array
- Some uploaded images are large in size → causing slowness, need to resize before processing