Детектирование и распознавание автомобильных номеров

Антон Зайцев

Введение

Цель: Разработка системы автоматического детектирования и распознавания номерных знаков автомобилей на видеопотоке.

Задачи:

- Детекция номеров на изображении
- Распознавание символов
- Обработка нескольких видеопотоков
- Создание десктопного приложения

Используемые технологии

YOLO (ultralytics) - Детектирование объектов на изображении

OPENCV - чтение видеопотока и обработка изображений

BYTETrack - трекинг объектов на видео

ONNXRuntime - оптимизация инференса моделей

PyQT - графическая оболочка приложения

SqlAlchemy - ORM для подключения базы данных

DVC - версионирование датасетов

Poetry - управление зависимостями проекта

Docker - изолирование среды для обучения

Pyinstaller - сборка исполняемых файлов

Архитектура системы

[Камеры] → [Предобработка] → [Детектирование номера] →[Трекинг]→ [Распознавание символов] → [OCR постобработка] → [БД] → [Интерфейс]

- Камеры/видеопотоки : источник данных
- Предобработка : нормализация, уменьшение размера
- Детектирование : определение области номера
- Распознавание : преобразование изображения в текст
- OCR постобработка : фильтрация, проверка формата, выбор наиболее вероятного варианта
- База данных : хранение результатов распознавания
- Интерфейс : вывод информации пользователю

Детектирование номеров

Для детектирования используется модель YOLOv11 Nano. Подходящая для работы в реальном времени и на мобильных устройствах.

Обучалась на датасете с HuggingFace

Плюсы:

- Высокая точность детектирования
- Низкие требования к ресурсам
- Различные варианты по сложности
- и точности (small, large)

Репозиторий проекта:

https://github.com/AntoxaZ18/lpr_detect

Абляционный анализ для YOLO

Исследование влияния различных параметров на производительность модели YOLOv11 Nano. разрешение 640*640 если не указано иное

Модель	val mAP50	val mAP50-95	GFLOPS	FPS(cpu)
YOLOv11 Nano original	0.986	0.822	6.5	27.9
блоки CSPA, C3K2 по одному	0.986	0.804	5.4	28.4
c2spa->c3k2 1024->768 512->384	0.986	0.804	5.1	31.9
SPFF->Maxpool	0.985	0.792	4.5	32.4
768->512, 256->192 384->256	0.985	0.785	4.2	35.8
c3k2->conv	0.985	0.769	4.0	38.7
320*320 (NWD loss)	0.966	0.742	4.0	80+

ByteTrack для трекинга номеров

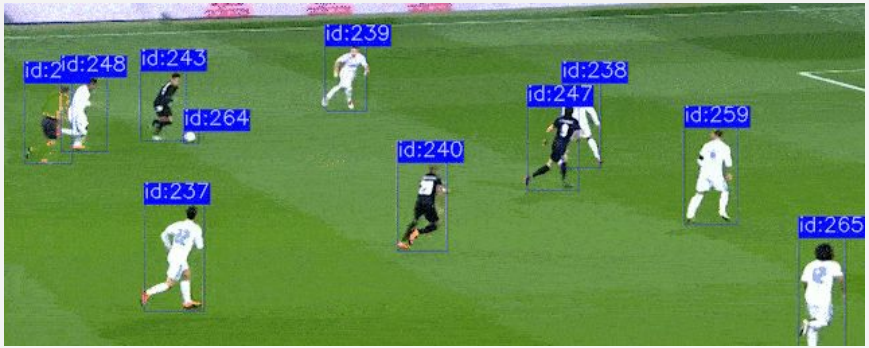
Алгоритм трекинга объектов, сочетающий детекцию и трекинг в реальном времени, который оптимизирован для скорости

Роль:

- связывает обнаруженные номера между кадрами
- повышает стабильность работы при движении автомобиля
- уменьшает вероятность потери номера при смещении или частичной маскировке

Плюсы:

- Хорошая точность трекинга
- Низкие требования к ресурсам
- Поддержка работы с видеопотоком



Модель LPRNet

- Специализированная модель для распознавания номеров
- Использует CNN архитектуру для последовательной обработки символов
- обучена на датасете с платформы hugging face

Преимущества:

- Высокая точность распознавания номеров (acc 0.95, char acc 0.98+)
- Поддержка разных форматов номеров (К сожалению в датасете не было квадратных номеров и спецномеров =( )

- Невозможность сконвертировать в последний ONNX opset. (замена слоев + переобучение)

Репозиторий проекта: https://github.com/AntoxaZ18/lprnet_ocr

Spatial Transformer Network

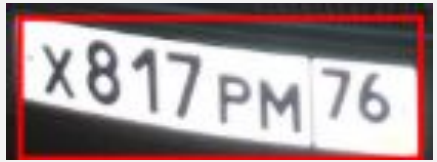
Позволяет модели обучаться преобразованиям изображения (поворот, масштабирование). Обучается совместно с lprnet

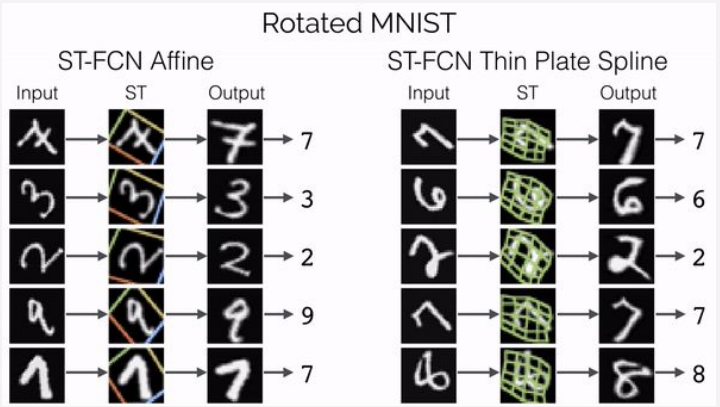
Роль в проекте:

- улучшение качества изображения перед распознаванием
- автоматическая коррекция ориентации номера

Преимущества

- устойчивость к искажениям
- упрощает работу LPRNet





Версионирование и управление моделями

- Модели хранятся в формате onnx с названиями, включающими версию и параметры: yolo_lprnet_v1_640x640.onnx
- Поддержка разных размеров изображения модели
- Используется файл конфигурации для выбора нужной модели
- Поддерживается переключение между версиями без изменения кода
- Различные варианты загрузчиков позволяют загружать модели с репозиториев (S3) или локального хранилища*

*реализовано только из локального хранилища

Масштабируемость и многопоточность

- Система поддерживает работу с несколькими видеопотоками (запуск несколько сессий для разных моделей)
- сбор батчей из разных видеопотоков
- использует механизм futures и пула потоков
- оптимизирован при помощи onnxruntime
- возможность использование ускорителей
- адаптивная очередь на входе для балансировки нагрузки
- несколько видеопотоков в реальном времени* ~100FPS (Ryzen 5 3600)
- работает как на “слабых” так и производительных процессорах

Приложение

Фреймворк PyQT

Используется pyinstaller для сборки

Поддержка нескольких ОС (WIN)

Плюсы:

- легкость развертывания
- независимость от python окружения

Репозиторий проекта: https://github.com/AntoxaZ18/lprecognition

Заключение и перспективы

Разработана работающая в реальном времени система детектирования и распознавания номеров с несколькими видеопотоками. Результаты распознавания сохраняются в базу данных для последующей аналитики.

Возможные улучшения:

- Использование других моделей для детектирования для ускорения либо более мощных для работы в сложных сценах

- Перенос инференса на GPU

- Клиент-серверная архитектура

- Интеграция с Gstreamer

- Мониторинг/логгирование (Prometheus/Grafana)

- API