PWC Data-lympics 2019

Challenge 1:

Car Registration Number Recognition API (CRNRA)

HELLO!

We are CityU Apps Lab!

Real World Values

Real-time law enforcement

Expired license plates

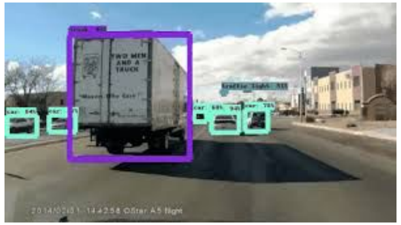
With Car Registration Number Recognition, penalties could be applied to the drivers with a expired license plate easily.

License plates captured would be sent to the database and check whether they are expire or not.

Tracking of Suspicious cars

With Car Registration Number Recognition, malicious suspect could be tracked by real time based on their license plates and their exposed locations.

A large clusters of advanced security cameras could be set all around a city. Therefore a specific car would be detected anywhere.



Automatic Parking and restricted zone management

Automatic Parking

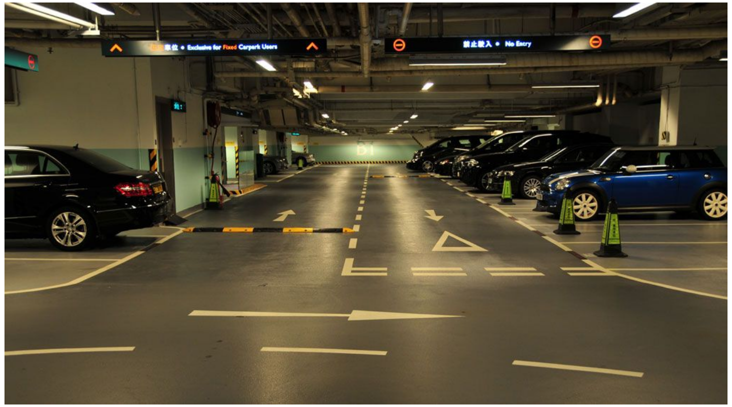
With Car Registration Number Recognition, process of parking for cars could be automated.

License plates are captured so that billings or charges could be sent accordingly.

Restricted zone management

With Car Registration Number Recognition, entry of malicious cars which are not belonging to a trusted list would trigger alerts to the securities.

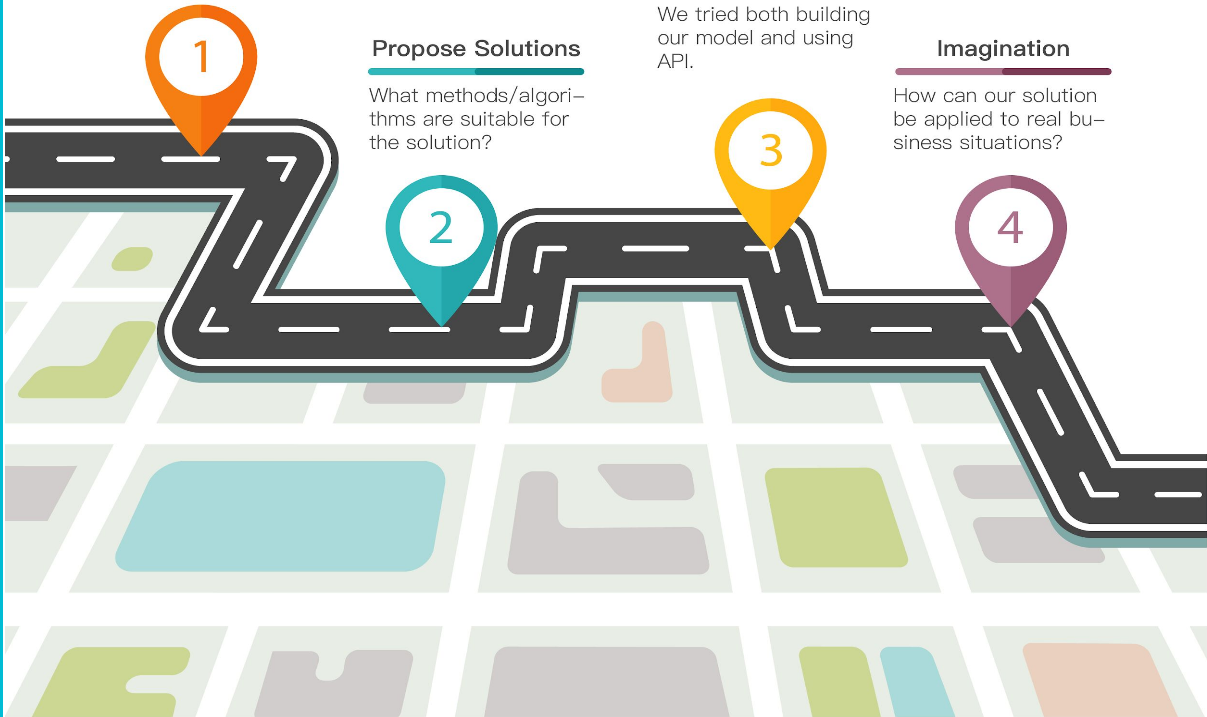
License plates are obtained and stored so that entry records of all identities are traceable.



Adding value to businesses

- Automating and thus reducing the need for human effort.
- Reduces the risk of error through deployment in IoT.
- Improved security and digitization.

Roadmap



Problem Identification

What challenges do we need to comeover?

Propose Solutions

What methods/algori–thms are suitable for the solution?

Implementation

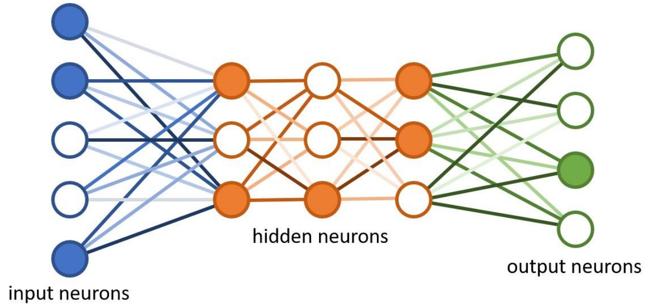
We tried both building our model and using API.

Imagination

How can our solution be applied to real bu–siness situations?

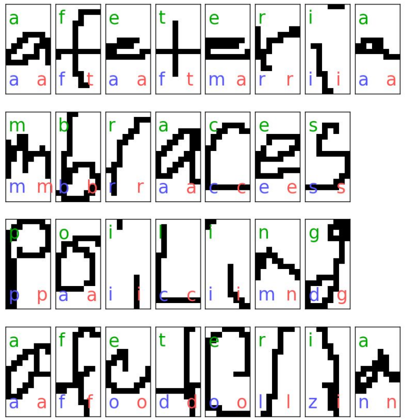
Why Faster RCNN?

- Readily supported across Deep Learning frameworks.
- Suitable for realtime tracking.
- High accuracy!



Optical Character Recognition (Bidirectional GRU)

- openalpr benchmark for training data
- Insufficient Hong Kong number plates
- Overall performance quite poor



Final SOLUTION

1) An Easy-to-Use API

Applicable

Can be applied in majority of programming languages

Similarity

Similar to other APIs
(Can export JSON)

=> Easy to learn

2) Simple Embedment Procedure

Use the API tokens to call requests from our backend server

$\longrightarrow$

Integrate into development environment with own camera library

3) Accurate Recognition

1. Can adjust frame rate
2. Using various recognition algorithms

Limitations and Potential Problems

- Dependency to third-party API and library
  - Performance would be vary if dependencies change

Technical Detail &

DEMO

Python Django

- Build the backend service in 3 hours

- Created 3 api endpoints

1. Process the data
2. Display the data (without any filtering)
3. Filter the data to the specified format

Q&A Session

Thank you for listening!