LOW LEVEL DESIGN

Vehicle Number Plate Detection

Created by Vikram Singh

GitHub Profile : VkasRajpurohit (github.com)

Document Version Control :

Date issued	Version	Description	Author
May 26th, 2022	1	Initial LLD V1.0	VIKRAM SINGH

TABLE OF CONTENTS

Chapter	Page No.
Abstract	3
Overview	4
1. Logging	4
2. Process Flow Detail	4
2.1 Problem Statement	5
2.2 Get Dataset & Validation	5
2.3 Data Preprocessing	5
2.4 Split Dataset	5
2.5 Model Building	5
2.6 Save Model	5
2.7 Create web application | User I/O workflow	5
2.8 Deployment	6

ABSTRACT

The purpose of this LLD (Low Level Design) document is to give the internal logical design of the actual program code for Vehicle Number Plate Detection. Low-level design is created based on the high-level design. LLD describes the class diagrams with the methods and relations between classes and program specs.

It describes the modules so that the programmer can directly code the program from the document. The code can be developed directly from the low-level design document with minimal debugging and testing. Other advantages include lower cost and easier maintenance.

Overview:

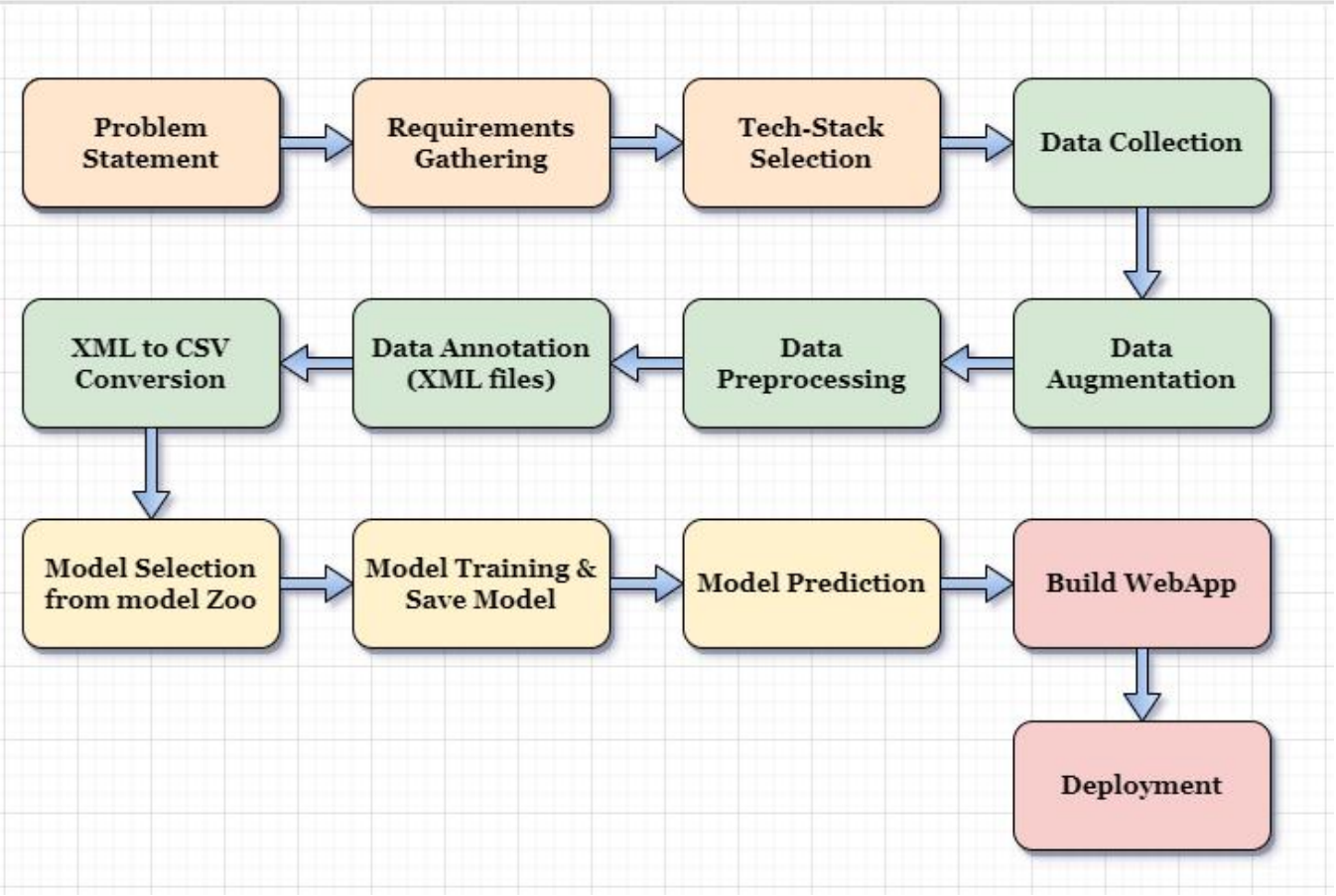
Title	:	Vehicle Number Plate Detection
Domain	:	Traffic Surveillance and Security
Tools & Technology	:	Python | OpenCV | PyTesseract | Tensorflow | Keras | Image Data-Processing | Deep Learning | Flask-API | HTML | Bootstrap | GitHub | AWS
Tensorflow Model	:	Inception-ResNet-v2
IDE	:	PyCharm | Google Colab

1. Logging:

Model should be able to log every activity done by the user.

- Logging is mandatory for easy debug issues.
- The system should be able to log each and every system flow.

2. Process Flow Detail:



2.1 Problem Statement:

Build a solution that should be able to get an input as vehicle image and provide the result as extracted number from vehicle number plate.

2.2 Get Dataset & Validation:

For the Problem statement, collected image data from different-different sources i.e. collected by web scraping (didn't got data as expected), then collected data manually by recording & clicking pictures on traffic signal, also a part of dataset collected via open-source data.

2.3 Data Preprocessing:

In this, data preprocessing will be handled for the entire image data. In this data preprocessing need to perform below details -

- Data Augmentation to increase dataset size.
- Data Resize.
- Data Annotation.
- XML to CSV Conversion.

.

2.4 Split Dataset:

Split the dataset into train data & test data.

2.5 Model Building:

We will use the **Inception-ResNet-v2 model** with pre-trained weights and train this to our data. Visualize the result with TensorBoard.

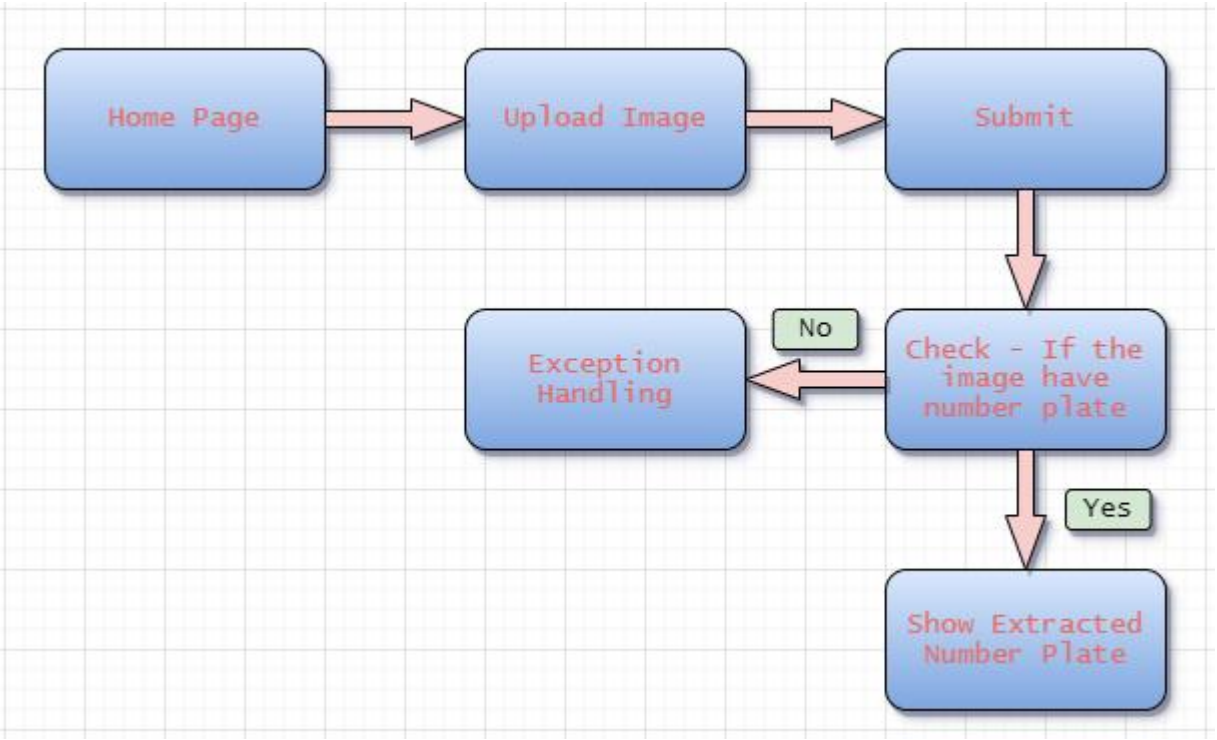
2.6 Save Model:

After training model saved as **model_number_plate.h5**. Saved model can be used later. Now the training part has been completed.

2.7 Create web application:

Create a User Interface web application using Flask-API & HTML. Model prediction part will be done here. Need to create pipeline for prediction part for all the steps done in the training part i.e. data processing, re-sizing data, create bounding-box & extract numbers from bounding box. Use saved model for prediction of the user input image data.

User I/O workflow:



Home page is the landing page, it will ask for the user input. User will upload the vehicle image & hit submit. If image have number plate then show result as extracted number plate, however if the input image do not have number plate or incorrect input image given then handle exception accordingly.

2.8 Deployment:

Now model is ready for productionisation. For model deployment use **Heroku** or **AWS.**

---------- **End of LLD** ----------