VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Computer Science and Engineering

J Component report

Programme : **B.Tech (CSE)**

Course Title : **IoT Domain Analyst**

Course Code : **ECE3502**

Slot : **A1**

Title: Smart Traffic Monitoring System

Team Members: **Avihrik Basak | 19BCE1202**

Pranish Shrestha | 19BCE1758

Faculty: SHOLA USHA RANI **Sign:** S.Usharani

Date: 26/4/22

Abstract

Traffic jams are a rising issue in countries like India, which has numerous densely populated cities and a majority of the population has access to vehicles like cars and bikes. These lead to traffic congestion as the city's locomotive architectures, ie., roads, highways, can't keep up with the rate at which the density of traffic increases. While traffic monitoring authorities try their best at solving such issues by implementing rules to break up traffic flow and let them flow smoothly, this requires for the people to follow these rules. Our proposed system lets us record the number plate of the vehicles that ignore the traffic lights and continue moving despite the traffic light being red. This system will make sure people follow this rule, or atleast, think twice before breaking it as their number plate will be stored in the traffic police database.

Content

Introduction

Traffic rules are an essential step at managing the rapid growth of vehicles in densely populated cities. Without them, the intensity of traffic accidents will start building up and cause more injuries and loss of lives. A survey from WHO revealed that approximately 1.3 million lives were lost from traffic accidents. And around 50 million received non-fatal injuries from traffic accidents. Not to mention the property damages these accidents cause which costs a lot of money to repair.

Evaluating these effects of traffic accidents, we are able to realise the scale of dangers they possess. As such, we must come up with ways to minimize these incidents. And one of the best ways to do so is by creating a system that reinforces a pre-existing system of managing traffic issues.

Our system consists of a “deep neural network”-based program that extracts the licence plate of vehicles from a video feed and then translates the picture into texts for efficient storage of the licence plate data. This data is then collected and stored in a table format (xlsx, csv) so that the authorities can take proper action against the people who break traffic light laws.

Literature Survey

- The importance of mandatory traffic rules have been discussed in the paper "Traffic rules and traffic safety" [1]. The authors, after their research and data collection work, had come to the conclusion that a huge percentage of drivers, who were caught in accidents, were aware of traffic rules but didn't bother to follow them as they didn't realise the importance of rules at keeping everyone safe. Thus, enforcing traffic rules are a necessity when dealing with traffic safety.
- "Smart Traffic Light Control System" [2] and "Smart Traffic Management System Using Internet of Things" [3] both highlight the need and beneficial effects of automated and IoT-based systems have on the traffic flow.
- Prashant Jadhav, in their paper [4], have mentioned the use of a MATLAB-based algorithm, in conjunction with surveillance cameras to detect the intensity of traffic on a particular road and give it a priority-based clearance on the intensities of nearby roads. However, that system turned out to be too bulky to see wide usage.
- In "Automatic Number Plate Detection for varying Illumination Conditions" by Rachna Boliwala and Manish Pawar [5], the authors have used various image processing and edge-detection algorithms to detect the region of the licence plate. However, the use of these traditional techniques yield inefficient results when compared to deep learning models.
- X. He and L. Deng's "Deep Learning for Image-to-Text Generation: A Technical Overview" discusses the various advances made in the field of image-to-text conversion using deep learning techniques and also their short-comings. We also got to know about other applications of deep learning that are related to image analysis.

Modules

1. Dataset acquisition

For this project, we acquired our dataset from kaggle (https://www.kaggle.com/datasets/andrewmvd/car-plate-detection). This dataset consists of 433 images of number plates from all around the world and also their annotations which includes the specific coordinates of the number plate in that image, which helps us train the model to pinpoint where the number plate is located.

After acquiring the dataset, we split up the images into training and testing partitions in the ratio of 75:25. The training partition is going to be what our object detection model is trained on, and the testing partition is what our object detection model is evaluated on.

2. Training our model

In order to detect number plates we need to first train the object detection model. This allows us to detect number plates from an image, video and in real time.

We use pretrained models from Tensorflow Model Zoo and use it for transfer learning. Transfer learning means to reuse a predeveloped model and make changes to it depending on our needs. After making changes to this model we take the images from the training dataset and feed it to the model, and train it for 10000 steps to get a more accurate model.

Libraries used:

a) Tensorflow: It is an “open source” machine learning and artificial intelligence library, using data flow graphs to build models. It allows developers to create large-scale neural networks with many layers.

b) TensorFlow Object Detection API: It is an open source framework built on top of TensorFlow that makes it easy to

construct, train and deploy object detection models. It is used for predictions and creations.

3. Testing

We can now leverage the trained model to detect the number plate from an image, recorded video and in real time.

Libraries used:

a) Optical Character Recognition: OCR gives us the ability to extract text from the region of interest and work out what's being rendered.

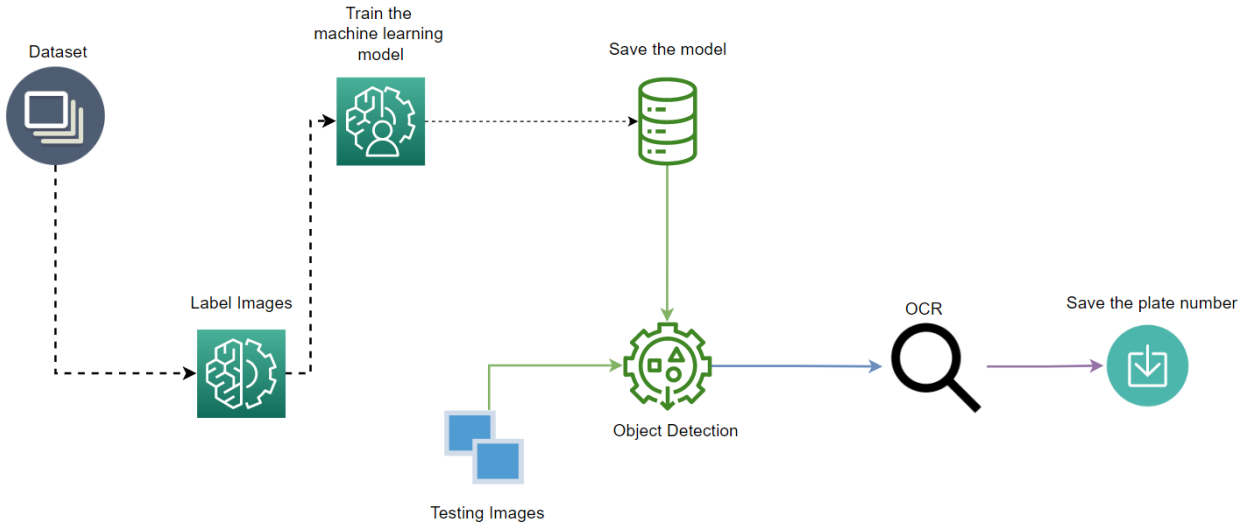
b) OpenCV: It is a great tool for image processing and performing computer vision tasks. It is an open-source library that can be used to perform tasks like face detection, objection tracking, landmark detection, and much more. It is used here to read an image, video, and capture real time videos through our webcam.

c) matplotlib: This library is used to plot/display the image captured by the system.

4. Saving the results

After testing our trained model with several images, recorded videos and real time video we save the cropped image of the detected plate and the extracted plate number in the designated folder and a csv file respectively.

Architecture



Firstly we collect imageset from kaggle then we will label each and every image of the vehicle. This process is very time consuming and is an important step for the project. Using the dataset we train the machine learning model for object detection, i.e. for detecting or recognising the number plates. Once the model is trained we save the progress of the model as checkpoints, which we will later use to evaluate the model using test datas.

We now take test images and videos, and restore the saved model and pass the image to the restored model. Now this model returns the detected object i.e. position of the number plate, which can also be labelled as region of interest (ROI).

Using Optical Character Recognition(OCR) and the object that has been detected from our model, we extract the text present in the image. Here we use EasyOCR to perform this process.

Now finally, after detecting the position of the number plate and the plate number, we will save the data in csv format.

Experimental setup

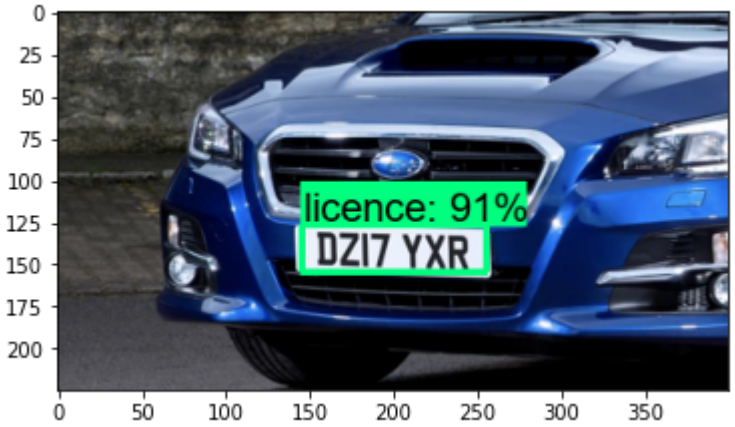
For Implementation of the project, we have used/worked with:

- Python (in IDE – Jupyter Notebook)
- For Handling Data Storage (in arrays): NumPy package for large vector operations
- For Image Vision: library cv2 (OpenCV)
- For building our models for the project: TensorFlow
- For recognising text from an image: EasyOCR
- For dealing with filenames, paths, directories: os

For our project we have performed several experiments in total with three different setups:

1. **Detect from an Image:**

   In this setup we take an image of a vehicle with a number plate and pass it through our trained model using the **'imread'** function of **cv2** library. The model will then detect the position of the number plate and store the detection points. The image is then displayed using the **'imshow'** function of **matplotlib** along with the accuracy percentage of detecting the region of interest.



   These points are then passed through a function in which ROI is filtered and only the number plate is retrieved. The number plate then goes through the process of OCR where the text is detected from it.

   The detected plate and the plate number is then saved in the 'Detection_Image' folder and detection_result.csv',

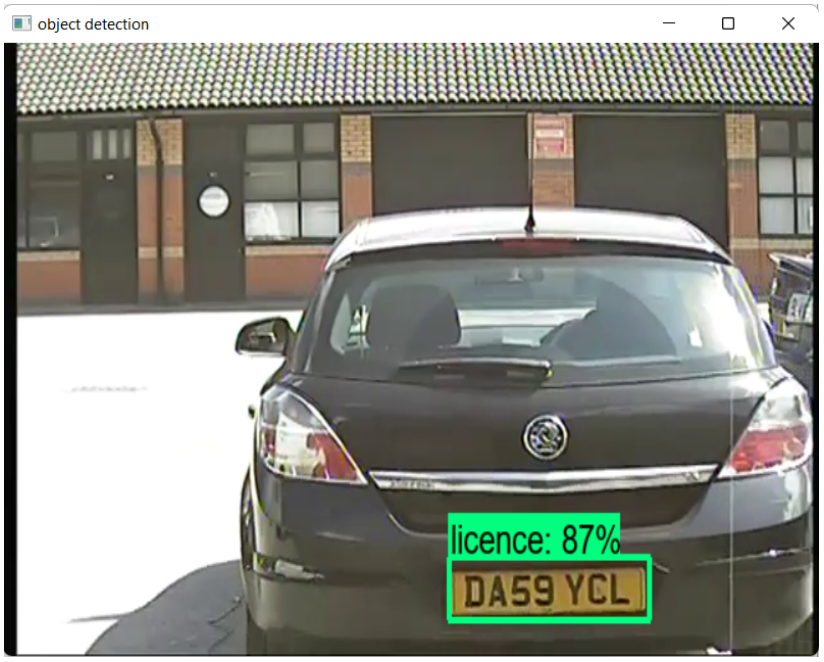
```
[0.56524885 0.36464155 0.6860421  0.6403215 ]
[127.18099058 145.85661888 154.35946584 256.12859726]
[([[4, 0], [106, 0], [106, 27], [4, 27]], 'DZI7 YXR', 0.8476270321759378)]
```

This output is still not refined, so we pass it through another function to filter out only the OCR.


```
['DZI7 YXR']
```

2. Detect from a recorded video:

In this setup we pass an already recorded video through the model using the **'VideoCapture'** function of **cv2**. The model captures each and every frame of the video and detects the position of the number plate and displays the accuracy percentage of detecting the ROI.



The frame captured is then passed through the OCR function to retrieve the text in the number plate.


```
CUDA not available - defaulting to CPU. Note: This module is much faster with a GPU.
['DA59YCL']
```

The detected plate and the plate number is then saved in the 'Detection_Image' folder and 'videoresult.csv'.

3. Real Time Detection from Webcam

This setup is the main part of our project where we implement the traffic light system to keep record of the number plates of vehicles that disobey the traffic rules and jump a red light.

We make use of the '**VideoCapture**' function of **cv2** to take control of our webcam and record the frames. There are 3 scenarios in this setup:

a. When the traffic signal is green:

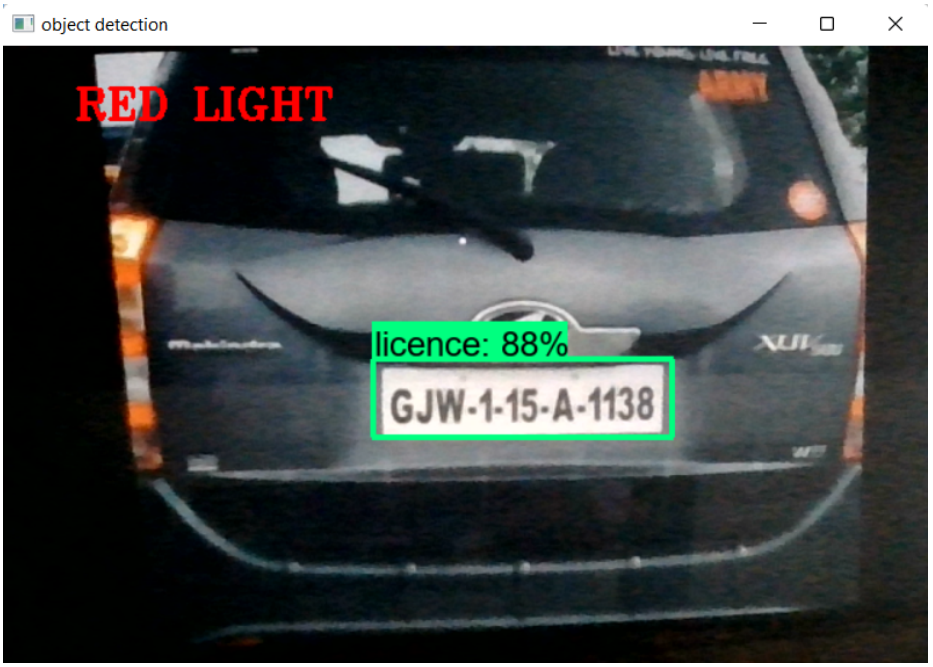
Since the traffic signal is green, the vehicle is not breaking the rules so the number plate of the vehicle will not be captured and detected.

b. When the traffic signal is yellow:

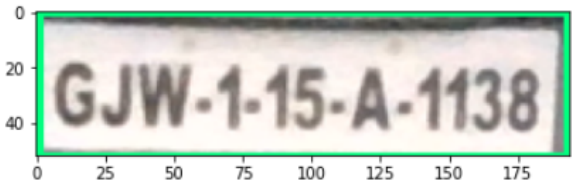
Similar to the green signal, when the traffic signal is yellow, the vehicle is not breaking the rules so the number plate of the vehicle will not be captured and detected.

c. When the traffic light is red:

Our traffic monitoring system will now start capturing and detecting the number plates of vehicles that disobey the traffic signal and decide to jump a red light.



The captured frame is then sent to the ‘ocr_it’ function where the number plate region is filtered and OCR is applied to it to extract the text from the image. The ROI and the detected plate number is printed and saved in a folder and appended to a csv file.




```
CUDA not available - defaulting to CPU. Note: This module is much faster with a GPU.
['GJW-115-A.1138']
```

The result is saved in this format: image_file.jpg, detected plate number.

Results and discussion

The model is trained for 10000 steps and with every 1000 steps, a checkpoint is created where a trained model is stored that we can later load and work on it.

Testing the same image with different checkpoint models, we get the following results;

checkpoint-11:

checkpoint-10:

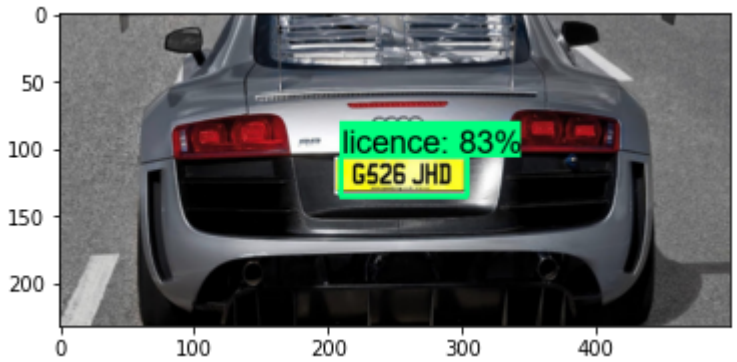
checkpoint-9:

checkpoint-8(currently being used):

checkpoint-7:

checkpoint-6:

checkpoint-5:



Putting the accuracy percentages in a tabular format:

Checkpoint	Detection Accuracy
5	83%
6	83%
7	89%
8	95%
9	93%
10	92%
11	91%

Line graph:

"**Overfitting**" is a modelling error in statistics that occurs when a function is too closely aligned to a limited set of data points. As a result, the model is useful in reference only to its initial data set, and not to any other data sets.

From the above data, we can come to a conclusion that when the model was run for 10000 steps overfitting occurred, making the later models less accurate.

Result from the experiments performed:

6109776c-c4cc-11ec-8e3c-00d861087345.jpg	['MH20cs 1940,']
7bda32bc-c4cc-11ec-b423-00d861087345.jpg	['MH 20cS 1941']
bcff8824-c4cc-11ec-93ce-00d861087345.jpg	['AHIZ DE1433']
be5e9150-c4cc-11ec-b965-00d861087345.jpg	['MHIZ DE1433']
bfc4203b-c4cc-11ec-944a-00d861087345.jpg	['IHIZ DE1433']
c12c72bc-c4cc-11ec-8fba-00d861087345.jpg	['IHI2 DE1433']
c2984b0e-c4cc-11ec-bd4b-00d861087345.jpg	['IHI2 DE1433]']
0a275fb4-c4cd-11ec-8864-00d861087345.jpg	['HR 26 DA 04741']
0b848e81-c4cd-11ec-81d9-00d861087345.jpg	['HR 26 DA 0471/']
475c344e-c4cd-11ec-b4ea-00d861087345.jpg	['3 CAY 9324']
5ba78a43-c4cd-11ec-8cf3-00d861087345.jpg	['DLAchL9374']
5e88a3d0-c4cd-11ec-84a9-00d861087345.jpg	['DL 3 CAY 9324']
6999e084-c4cd-11ec-a994-00d861087345.jpg	['GJW-1.15-4.1138 |']
6af8e886-c4cd-11ec-b957-00d861087345.jpg	['GJW-1.15-4.1138']
805af1cb-c4cd-11ec-9127-00d861087345.jpg	['HRSLX9099']
81baa9b7-c4cd-11ec-8f0b-00d861087345.jpg	['HRSLY 9099']
8680ccb6-c4cd-11ec-bd23-00d861087345.jpg	['HHILEU3498']
892744b2-c4cd-11ec-a997-00d861087345.jpg	['HH14EU3498']
8bbee183-c4cd-11ec-8308-00d861087345.jpg	['HH14EU3498']
8d16d45c-c4cd-11ec-8376-00d861087345.jpg	['HH1LEU3498']
8e6f2682-c4cd-11ec-b1d9-00d861087345.jpg	['HHILEU3L98']
8fd367c5-c4cd-11ec-a2c4-00d861087345.jpg	['HH1LEU3498']
91350344-c4cd-11ec-add7-00d861087345.jpg	['HH1LEU3498']
994b82a2-c4cd-11ec-87a4-00d861087345.jpg	['K404HN3622']
9aa1607e-c4cd-11ec-b19d-00d861087345.jpg	['K404HN3622']
70f3d182-c72d-11ec-ae10-00d861087345.jpg	['GJW-115-A.1138']
72751519-c72d-11ec-add6-00d861087345.jpg	['GJw-1.15-4-1138']
76b5d773-c72d-11ec-aef2-00d861087345.jpg	['GJW-1.15-4.1138]']

realtimeresults.csv

The results obtained from the real time webcam setup are stored in the 'realtimeresult.csv' file, which can be used by the traffic police department to take necessary actions.

Hypothesis from the results

From collected data and outputs, we are able to observe the efficiency at which the deep-learning model is able to extract licence plate information of people who break traffic laws. As such, we hope our system can be feasibly applied on traffic lights as an add-on system in a cost-effective manner. And that it will reduce the chances of people running over a red light and breaking other traffic related laws as people are less willing to break a rule when someone is observing them.

Future Works

1. Integrate the project with IR sensors to detect the speed of the vehicle, and record the plate number of vehicles that are speeding above the speed limit.

2. Since covid is on the rise again, we could integrate our number plate recognition system with face mask recognition system and record the plate number of vehicles whose passengers are not wearing masks.

3. Integrate a system that cross-checks the licence plate number with the police database to check for fake numbers or repeating offenders.

Conclusion

The use of deep-learning model, via the TensorFlow library, and EasyOCR has allowed us to create a licence plate recognition system that allows the traffic officials to record the licence plate info of any vehicle that runs through a red light. This reduces the occurrences of people trying to break traffic laws and thus, in turn, reduce the chances of traffic accidents. And since the algorithm stores only the licence plate image and licence plate number in text format, we can use a high resolution surveillance camera without the worry of running out of storage.

References

1. L Åberg, *Traffic rules and traffic safety*, Safety Science, Volume 29, Issue 3, 1998, Pages 205-215, ISSN 0925-7535,
2. B. Ghazal, K. ElKhatib, K. Chahine and M. Kherfan, "Smart traffic light control system," 2016 Third International Conference on Electrical, Electronics, Computer Engineering and their Applications (EECEA), 2016, pp. 140-145
3. Javaid, Sabeen & Sufian, Ali & Pervaiz, Saima & Tanveer, Mehak. (2018). Smart traffic management system using Internet of Things. 393-398.
4. P. K. K. P. S. T. Prashant Jadhav, "Smart Traffic Control System Using Image Processing," International Research Journal of Engineering and Technology (IRJET), vol. 3, no. 3, pp. 2395-0056, 2016.
5. R. Boliwala and M. Pawar, "Automatic number plate detection for varying illumination conditions," 2016 International Conference on Communication and Signal Processing (ICCSP), 2016, pp. 0658-0661
6. X. He and L. Deng, "Deep Learning for Image-to-Text Generation: A Technical Overview," in IEEE Signal Processing Magazine, vol. 34, no. 6, pp. 109-116, Nov. 2017
7. Jun-Wei Hsieh, Shih-Hao Yu, Yung-Sheng Chen, “Morphology- based License Plate Detection from Complex Scenes”, 16th International Conference on Pattern Recognition (ICPR'02) Vol 3, 2002.
8. Kwang In Kim, Keechul Jung and Jin Hyung Kim, “Color Texture- based Object Detection: an Application to License Plate Localization”, Lecture Notes in Computer Science, pp. 293-309, Springer, 2002.
9. Yasuharu Yanamura, Masahiro Goto, Daisuke Nishiyama, “Extraction and Tracking of the License Plate Using Hough 9 9 50 S Possible overlapped windows 63 Transform and Voted Block Matching”, IEEE IV2003 Intelligent Vehicles Symposium Conference, 2003.
10. [6] A. Rahman, Ahmad Radmanesh, “A Real Time Vehicle’s License Plate Recognition”, Proceedings of the IEEE on Advanced Video and Signal Based Surveillance, AVSS’03, 2003.
11. K. Kanayama, Y. Fujikawa, K.Fujimoto, and M.Horino, “Development of Vehicle-License Number Recognition System Using Real-Time Image Processing and Its Application to Track-time measurement”, In Proceedings of IEEE Vehicular Technology Conference, pp798-804, 1991.
12. C. Arth, C. Leistner, and H. Bischof. TRIcam: An Embedded Platform for Remote TrafficSurveillance. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (Embedded Computer Vision Workshop), 2006.

13. K.K. KIM, K.I., KIM, J.B. KIM, and H.J. KIM, Learning-Based Approach for LicensePlate Recognition Proceeding of IEEE Signal Processing Society Workshop, Vol. 2, pp.614-623, 2000.
14. S.H. Park, K.I. kim, K. Jung and H.J. Kim, Locating car license plate using Neural Network, Electronic Letters, Vol. 35, No. 17, pp. 1474 1477, 1999.
15. Smith R, "An Overview of the Tesseract OCR Engine," in IEEE Ninth International Conference Proceeding of Document anay and Recognition, 2007.