**License Plate Recognition**

Instructor: Assoc.Prof. Hai Pham Van

Students:

Nguyen Dinh Loc, Nguyen Thanh Nam, Ngo Quang Viet, Vu Dao Nguyen, Nguyen Manh Dung

School of Information Technology and Communication,

Hanoi University of Science and Technology, Hai Ba Trung, Hanoi, Vietnam.

**Abstract-** License plate recognition is not a new topic for researchers or real-life applications. Its applications can be found on many daily life systems such as smart parking lot with cameras installed, Vietnam’s highway electronic toll collection system (VETC), … But for a student studying AI, this can be a suitable topic to not only study about, but also a good and realistic model that can be implemented. This report considers the use of YOLO model to classify characters extracted from license plate images; the techniques used to extract the license plate segment from the images and to extract number from the segment extracted.

**Keywords**: Object Detection, Optical Character Recognition, YOLO

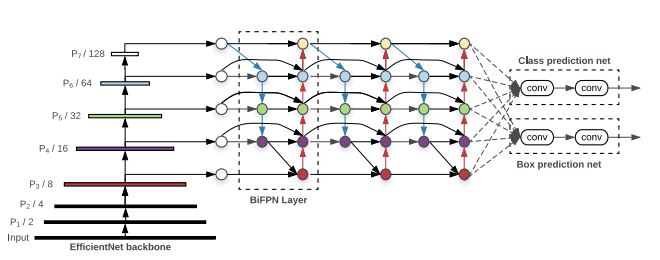
1. **Introduction**

Automatic License Plate Recognition (ALPR) is an important technology that has gained significant attention in recent years due to its various applications in fields such as traffic management, vehicle tracking, and crime investigation. In this report, we present a novel approach for license plate recognition using the YOLO (You Only Look Once) deep learning model. YOLO is a state-of-the-art real-time object detection model that has been widely used in various computer vision tasks. In the context of license plate recognition, the YOLO model is trained on a large dataset of license plate images to detect and recognize license plate numbers. The advantages of using YOLO for ALPR include its speed, accuracy, and ability to handle real-world scenarios with significant variations in lighting conditions, vehicle orientation, and license plate font styles. In this report, we aim to detect the region which contains license plate and read the characters, evaluate the performance of the YOLO model in license plate recognition. The experiment results show the effectiveness of the YOLO model in recognizing license plate numbers in various scenarios, and provide insights into the challenges and limitations of using YOLO for ALPR. Overall, the proposed approach provides a promising solution for license plate recognition and has the potential to be applied in a wide range of real-world scenarios. The rest of the report is organized as follows: Section 2 provides a proposed method in the field of license plate recognition, Section 3 describes the proposed approach, Section 4 presents the experimental results, and finally, Section 5 concludes the report and provides future directions for research.

1. **The proposed method**
   1. **Data Preparation:**

* License plate data are collected from the internet under the .jpeg type.
* Preprocessing the data: Extract the image data from the image.
* Split images data to train and test
  1. **Define YOLO v5 model:**

Object detection, a use case for which YOLOv5 is designed, involves creating features from input images. These features are then fed through a prediction system to draw boxes around objects and predict their classes.



*The anatomy of an object detector*

The YOLO model was the first object detector to connect the procedure of predicting bounding boxes with class labels in an end to end differentiable network.

The YOLO network consists of three main pieces.

* Backbone: A convolutional neural network that aggregates and forms image features at different granularities.
* Neck: A series of layers to mix and combine image features to pass them forward to prediction.
* Head: Consumes features from the neck and takes box and class prediction steps.
  1. **Train the model:**

**2.3.1 An Overview of YOLO Training Procedures**

The procedures taken to train a model are just as important as any factor to the end performance of an object detection system, although they are often less discussed. The two main training procedures in YOLOv5:

* **Data Augmentation**: Data augmentation makes transformations to the base training data to expose the model to a wider range of semantic variation than the training set in isolation.
* **Loss Calculations**: YOLO calculates a total loss function from the GIoU, obj, and class losses functions. These functions can be carefully constructed to maximize the objective of **mean average precision.**

**2.3.2 Data Augmentation in YOLOv5**

With each training batch, YOLOv5 passes training data through a data loader, which augments data online. The data loader makes three kinds of augmentations: Scaling, Color space adjustments, Mosaic augmentation.

**2.3.3 Auto Learning Bounding Box Anchors**

The idea of learning anchor boxes based on the distribution of bounding boxes in the custom dataset with K-means and genetic learning algorithms. This is very important for custom tasks, because the distribution of bounding box sizes and locations may be dramatically different than the preset bounding box anchors in the COCO dataset.

In order to make box predictions, the YOLOv5 network predicts bounding boxes as deviations from a list of anchor box dimensions.

Diagram, schematic

Description automatically generated

The most extreme difference in anchor boxes may occur if we are trying to detect something like giraffes that are very tall and skinny or manta rays that are very wide and flat. All YOLO anchor boxes are auto-learned in YOLOv5 when you input your custom data.

1. **Proposed approach**

**3.1 Reads the size of images in xml file and determine the coordinates of bounding box:**

**-** The information of the images include four edges coordinate points:

of the image: , which is stored in a csv file

**3.2 Labels the bounding box and split data:**

**-** We hold the information of bounding box in the form of two coordinate points**:** center\_x, center\_y and the size of bounding box: bb\_width, bb\_height.

center\_x =

center\_y =

bb\_width =

bb\_height =

Data train: 200 images

Data test: 25 images

**3.3 Train the model**

* Weights and hyperparameters: Take from Yolo v5
* Epochs: 100
* Batch size: 8
* Image size: 640 x 640
* Optimizer: Adam

The information of model in details you can find in this [link](https://colab.research.google.com/drive/1pQWwUDOWiOwzILHCwcP5y5UEyzvfzljf#scrollTo=LeEEn3YiVgBn):

**3.4 Load the image & model, generate the bounding box and generate the result of license plate:**

**3.4.1 Load the image & model:**

**-** We use the OpenCV library to load the image and our trained model.

****

*Test image*

* After this, image will be taken to the YOLO model to generate the bounding boxes.

**3.4.2 Generate the bounding boxes**

* The generated boxes from the model are redundant and undetermined, so we have to process the image in order to filter the bounding boxes and hold only one bounding box. In order to do this, we use two algorithms are: Non Maximum Supression (NMS) and Intersection Over Union (IOU).

So, with the IOU algorithm, we only focus on IOU index:

IoU is a parameter used to evaluate the overlap between two bounding boxes.

Suppose we have 2 boxes with the following information:

* Box 1 has the top-left coordinate , bottom-right coordinate .
* Box 2 has the top-left coordinate , bottom-right coordinate .

**Chart, box and whisker chart

Description automatically generated**

**In this case, IOU is calculated by the formula:**

**Chart, box and whisker chart

Description automatically generated**

**After that, Non Maximum Suppression is used.**

**Input: An array of bounding boxes, each box has the form** , in which:

* and are coordinates of top-left and bottom-right of bounding box, respectively.
* c is the confidence score of this bounding box, is returned from the YOLO model

The IOU threshold

**Output: An array of bounding box after remove the redundant boxes**

**The details of algorithm:**

**The symbols:**

* + **S**: bounding box under consideration
  + **P**: Set of input boxes of the algorithm
  + **thresh\_iou:** IOU threshold to remove redundant boxes
  + **keep:** Set of boxes after removing redundant boxes

**This algorithm includes 3 steps:**

**Step 1:** Select the box S with the highest confidence score in the set P, remove that box from the set P, and add that box to the keep set.

**Step 2**: Perform the IOU calculation between the box S just retrieved in step 1 and all the remaining boxes in the set P. If any box in P has an IOU with the box S under consideration that is greater than the threshold thresh\_iou, remove that box out of P

**Step 3**: Repeat step 1 until P has no more boxes.

After finishing the algorithm, **keep** contains all the boxes after removing the redundant boxes.

Chart, box and whisker chart

Description automatically generatedChart, box and whisker chart

Description automatically generated

Before and after use NMS

**3.4.3 Extract text after getting the bounding box (Optical character recognition):**

In this step, Pytesseract is used to read the characters in the bounding box, and return the results.

**A black car in a building

Description automatically generated with low confidence**

*Optical character recognition*

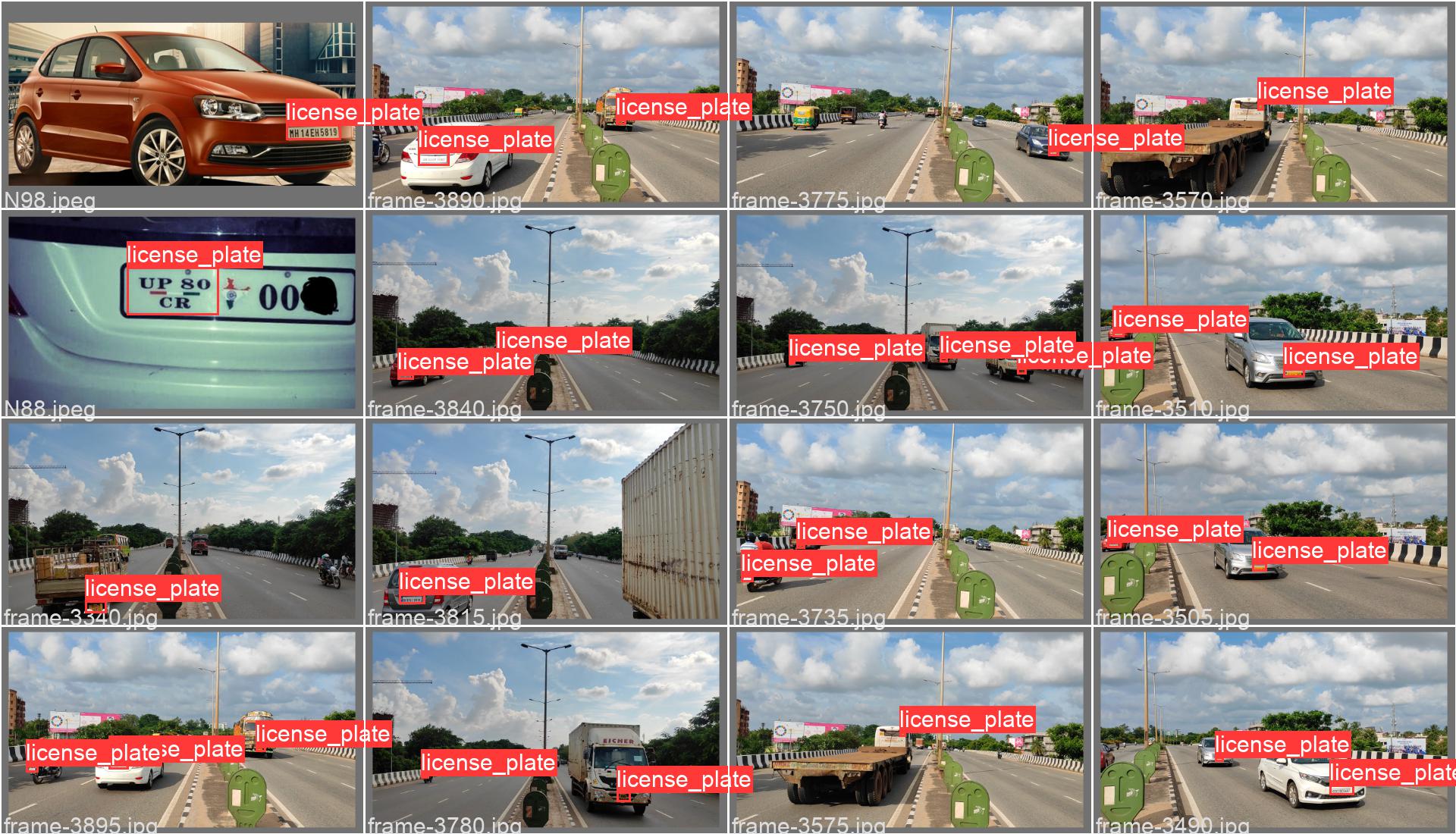
1. **Experimental results**

- In this report, we give some results to evaluate the performance of LPR model (YOLO v5 model). Besides the train and validation results, we use the precision, recall and mAP metrics.

The train and validation results:



*Train data of a batch*



Validation data of a batch

The performance of LPR model:

Graphical user interface, chart

Description automatically generated

*Precision, recall and mAP metrics*

* Precision = 0.9528
* Recall = 0.7017
* mAP = 0.8

- Precision - Confidence curve:

Chart

Description automatically generated

- Recall – Confidence curve:

Chart, line chart

Description automatically generated

- Precision – Recall curve:

Chart, line chart

Description automatically generated

Some results of our LPR model:



A picture containing text, car, screenshot

Description automatically generated

Graphical user interface

Description automatically generated

1. **Conclusions**

Based on our project, we conclude that deep learning-based models, particularly convolutional neural networks (CNNs), have shown promising results in LPR. Among CNN-based models, YOLO v5 stands out as a popular and efficient model for license plate detection. However, it is important to note that no single model or algorithm is perfect for all situations, and choosing the appropriate model depends on several factors, including the available data, processing power, and the specific use case.

Finally, we believe that the field of LPR will continue to evolve rapidly, with new techniques and models being developed to address the challenges faced by the industry. We hope that this project provides a useful overview of the current state of the field and helps researchers and practitioners make informed decisions when developing LPR systems.