Detection of Vehicular Number Plate System using Deep Learning Approach

[1]Shally Gupta, [2]Rajesh S. Singh, [3]Ratnesh Srivastava, [4]H. L. Mandoria

[1]M.Tech Student, [2]Assistant Professor, [3]Assistant Professor, [4]Professor & Head
[1]Department of Information Technology, College of Technology,
[1]Govind Ballabh Pant University of Agriculture and Technology,
Pantnagar, Uttarakhand, India.

Abstract: An automated system for recognizing vehicles' license plates are a growing need in order to improve security and for traffic control applications, particularly in major urban areas. Automatic Number Plate Recognition (ANPR) is a type of an Intelligent Transport System. While numerous studies on plate identification, character segmentation and character recognition have been performed, several challenges still remain. An efficient Vehicle Detection System is necessary to ensure traffic monitoring. In the last 4-5 years, several image processing and learning methods have been developed such as Optical Character Recognition (OCR) technique. The aspect of object detection, though, hasn't been exploited for ANPR framework in the previous researches focused on object detection. This research uses deep learning to leverage these object detection algorithms. You-Only-Look-Once (YOLO) and Convolutional Neural Network (CNN) have shown themselves to be the most efficient methods with regard to both supervised and unsupervised learning. This work studies several algorithms for object detection and deep learning, and compares their performance. The custom dataset was used to identify and recognize the license plate to ensure successful traffic management. We studied Deep Learning algorithms based on Image Segmentation using object detection algorithm through YOLO object detection technique in darkflow frameworks and character recognition based on CNN and also trained a model as base learner. This study establishes a method for real-time detection and identification of license plate, and drawing useful conclusions. The results of the simulation show that the deep learning methodology is more effective when detecting vehicle plates.

***IndexTerms* - Automatic Number Plate Recognition, Convolution Neural Network, Image Processing, Optical Character Recognition, License Plate, Deep Learning.**

I. INTRODUCTION

With the exponential growth in population that the world has witnessed over the last decade, the number of vehicles on our roads has also increased at a rapid rate, making the task of manually controlling and regulating the traffic way more difficult. License plate identification is a difficult task primarily due to differences in license plate styles and environmental factors such as harsh weather conditions, problem of reflection, headlight problems or blurred license plates [1]. For various applications such as automated toll collection, parking management, targeted ads, traffic monitoring and several others that require vehicle detection, robust number plate recognition architecture in different environmental conditions can be useful [2]. An efficient Vehicular Plate Detection System is required to ensure traffic management. Storing vehicle records is an essential part of any transport network and ANPR will prove to be infinitely more efficient than manual means of record keeping. Number plate identification usually has three levels – detection of license plate, segmentation of characters and recognition of characters. The earlier stages require higher precision or rather, near perfection, as failure to detect license plate would also likely result in failure in the further stages [3].

When working with several combinations of different approaches and thresholds, image processing methods do not very well scale up on a broad dataset for number plate segmentation. Therefore the implementation of YOLO (You-only-look-once), which is a CNN (Convolution Neural Network) based object detection algorithm, is preferred instead of conventional image processing technique [4].

Artificial Intelligence (AI) and Deep Learning (DL) have played a vital role in Intelligent Transportation Network in previous years. Hence this work aims to develop ANPR with a strong mixture of different object detection and DL algorithms. This provides us an idea about which method of detection offers the utmost precision. This work aims at identifying a vehicle's license plate and recognizing the characters on it. Hence, the license plate recognition system architecture should be implemented based on efficient and important algorithms for object detection.

The paper is structured as follows: A brief overview of the relevant work is addressed in section II. section III includes a data set overview accompanied by materials and methods in section IV, section V provides more information regarding the implementation details, section VI offers the discussion of results, and finally, the paper ends with the review of possible areas for future research in section VII.

II. LITERATURE REVIEW

Rasheed et. al. [5] developed a comprehensive license plate identification and recognition system focused on Hough Transformation by using Hough Lines and Matching Template Models. Detection was conducted using canny detector and the accuracy rate was 94.1 percent for vehicle plate extraction. Subhadhira et. al. [6] developed new Extreme Learning Method (ELM) recognition system. Using ELM classifier and Histogram of oriented Gradients (HOG), Thai license plate was used for preprocessing and extracting features. The system recognized plates consisting of both the identification number of car registration and the province with 89.05 percent accuracy. Liu and Lin [7] proposed a novel hierarchical character recognition scheme focused on supervised K-means and SVM to identify the blurred and tilted license plates. This system achieved 98.89 percent accuracy as compared to the state-of-the-art approaches to plate recognition, with an average increase of 3.6 per cent.

Selmi et. al. [8] used the first CNN model for LP detection and pre-processing measures to classify license plates and non-license plates in order to explore the ANPR system based on a deep learning approach. The second CNN model was used for the purpose of classification and recognition. The research was based on a tensorflow framework that utilized 37 classes of second CNN model. Lin and Li [9] then proposed the three-stage plate recognition system based on Mask-RCNN. For detection purpose YOLOv2 and for characters identification, Mask R-CNN was used. The findings identify vehicle number plates with bevel angles above 0-60 degrees and the mAP value of about 91 percent was further achieved. Suvarnam and ch [10] have proposed an ANPR method using CNN-GRU fusion, in which an optical character recognition technology is implemented to extract vehicle plates.

Shima [11] proposed an ANPR framework combined with morphological image processing and deep learning performing edge detection and Connected Component Analysis (CCA) for regional extraction. A pre-trained CNN "Alex-Net" was used as the feature extractor, showing a success rate of 89.7 percent. By using classifiers such as Random Forest Classifiers, Rep tree, IBK and K-star, Bhardwaj and Kaur [12] suggested a technique for number plate recognition. Effective accuracy was achieved by using K-star ML algorithms that show 99 percent accuracy in the data collection.

Mondal et. al. [13] developed an ANPR system beguiling CNN's learning feature skills. CNN's self-synthesized function was used because it distinguishes vehicle states that were organized in a feature detector echelon network. The findings were observed as 90 percent with less training samples with a higher precision rate. Lee et. al. [14] suggested an automated system to identify vehicle plate number, which had been implemented with a GPU. This research was carried out using NVIDIA Jetson TX1 boards and CNN-based plate recognition using AlexNet LP database showing accuracy rates of recognition as 95.24 percent.

III. DATASET DESCRIPTION

In this research, we are using deep learning algorithms to identify and recognize various license plates present in the custom dataset. A dataset of approximately 1371 license plate images was provided by the author for the custom dataset. OLYMPUS CAMEDIA C-2040ZOOM digital camera has been used to prepare a series of test images. The image archive includes photos of the rear view of vehicles, captured under various lighting conditions such as sunny, snowy, rainy, dusk, night-light, etc. The images are in the JPEG format [15].

Most of the vehicle images follow the Belgium’s standard format of license plate. Then it generates our own dataset by adding annotation to the images. We have created labels and annotation for custom YOLOv2 images dataset. Software named LabellMG tool is applied for annotation purposes. The data set is further split into the training dataset and the testing dataset. The percentage of images our model uses in the training is 90 percent images and the rest 10 percent is for testing. Videos are used in the testing phase for detecting and recognizing license plate.

Annotation on Images

Image annotation allows the computers to better understand the photographs and videos. This usually includes creating a periphery around important items with constructs such as bounding boxes in computer field, which makes the computer identify identical objects in the future.

In our project, annotation is done with the help of LabellMG Annotation tool called Labellmg.PyPi. LabelImg is a graphical image annotation method and a labelling object bounding box in images which is a simple tool for image labelling. It creates an XML for the image that can then be translated into CSV format using Python's XML Element Tree library and later on, it is fed to the Neural Network. Annotations are stored in PASCAL VOC format, the format used by ImageNet as XML files. Hence, 1371 image generation were done in the training and testing work while training by background process.

IV. METHODOLOGY

This section describes the different steps used in the analysis, accompanied by a brief description of the methods used to determine the outcomes. The proposed framework consists of two major stages. Firstly, the detection process is performed when the whole image frame is obtained as input. The detection process returns the detected plates as coordinates for bounding boxes. Secondly, the process of recognition takes the segmented plates and conducts the identification of characters. The entire procedure carried out can be visualized in Fig. 1.

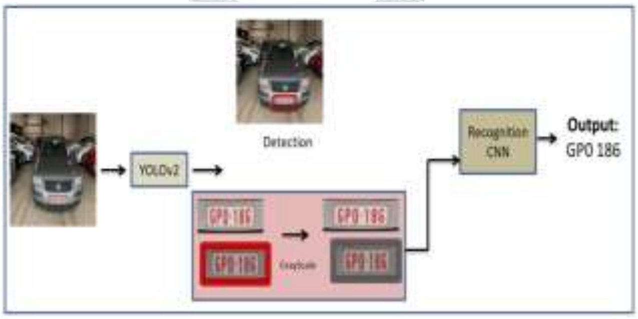

Fig. 1: Algorithm for License Plate Recognition using Deep Learning

A. License Plate Detection

We used a one-shot design technique of object detection network YOLO (You-Only-Look-Once), called YOLOv2, to perform license plate detection on number plate images. YOLO transforms the pictures into a single CNN in just one go, removing the need for regional and object proposals. YOLO's processing time is only 22ms per image, around 6-7 times quicker than Faster RCNN. The YOLO detection architecture as mentioned by author [16] is shown in Fig. 2. It consists of 24 convolution layers that are followed by 2 fully connected layers. Alternating 1×1 convolution layers decreases the space characteristics from preceding layers. Then the convolution layers are pre-trained to half the resolution 224×224 input image on the ImageNet classification mission, and then double the resolution for detection. YOLO's architecture is plain; it's just a neural convolution network. This neural network uses only regular types of layers such as convolution with a kernel of 3×3 and max-pooling with a kernel of 2×2. There is no fancy material in it. YOLOv2 has no fully connected layer in it.

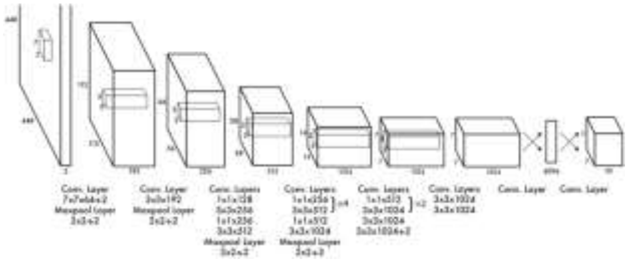

Fig. 2: YOLO Architecture

In 2017, YOLOv2 and YOLO 9000 were proposed by author [17] showing that for 67 FPS, YOLOv2 gives 76 percent MAP and at 67 FPS gives 78.6 percent MAP on the VOC 2007 dataset improved models such as Faster R-CNN and SSD. YOLO 9000 utilized architecture from YOLO v2, but was capable to detect over 9000 classes. YOLO's earlier version uses 224×224 as input size while training but the YOLOv2 version trains 448×448 for 10 epochs on ImageNet data in higher resolution. In spite of predicting coordinates immediately from the convolution network, YOLO make use of fully connected layers to predict bounding boxes but in YOLOv2, the fully connected layer is entirely omitted and then uses the anchor boxes to foretell the bounding boxes [18].

B. License Plate Recognition

For recognizing the number plate, Convolution Neural Network is used. The deep Learning CNN model takes the image cropped from the earlier step as input, whose dimensions were resized. Finally, the predicted Characters for every number plate are detected by using CNN.

1. Convolution Neural Network

The Convolution Neural Network or CNN, introduced in 1988, is a neural model of the network that incorporates the mentioned characteristics such as receptive fields, weight sharing and subsampling. It is a unique kind of multilayer perceptron trained using back propagation in a supervised way [19]. It is amongst the most popular computer vision machine learning systems and has attained state-of-the-art performance in processes such as recognition of characters, object recognition, face detection and pose estimation, voice recognition, license plate recognition, image pre-processing and segmentation processes [20].

Convolution Neural Networks are mostly used to process 2D images from the broad collection of models to learn high dimensional, nonlinear mappings. The CNN accepts an image input and transforms it with an adjustable weight 2D kernel. Sharing weight lowers the free parameters numbers. Convolution effects are applied altogether with an adjustable scalar, which is termed as a bias. Afterwards, the output is sent in an activation function that obtains 2D plane, known as the feature map. Then, every feature map is joined to a subsampling layer that decreases the size of the feature maps. Those subsampled feature maps are transferred through the Rectified Linear Network (ReLU), the normalization layer. The subsampling, convolutional, ReLU and normalization are stacked together, as depicted in the diagram shown in Fig. 3, and are regarded as the block.

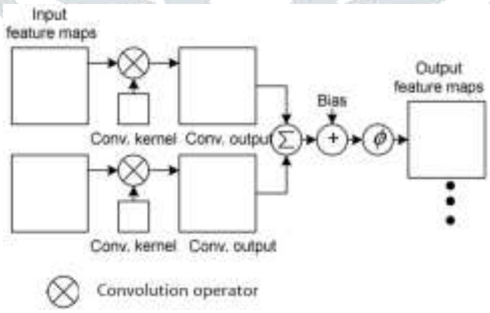

Fig. 3: Convolution Layer Example

2. CNN Layers

- The input image might contain thousands of pixels but convolution layers behaves as a feature extractor to extract features like corners, edges or endpoints by converting the input to weight-consistent kernels [21].
- ReLU is utilized as an input function for modelling the performance of a neuron, where f is the activation function of layer l to perform unit wise computations [22]. The Rectified Linear Network (ReLU) computes the f(x) =max (0, x) function, which increases the all over architecture's nonlinear properties.
- Subsampling offers an additional translational invariance that reduces deformation and noise. Pooling is performed by positioning the window at each location in the map of every feature and calculates single value per window for the map of the output feature, resulting in subsampling.
- Normalization tends to generalize the CNN. The output of the topical contrast normalization layer enforces local competition into a feature map between the adjacent units.
- In CNN architecture, the fully connected layer is the final layer. In this layer, all the feature map units from the preceding layer are fully connected to every unit in the current layer.

V. IMPLEMENTATION

The proposed system was implemented in a Visual Studio. The integrated development environment for Visual Studio is an innovative release pad for Python and several different programming languages that can be used to edit, debug, and test code, and further publish an app. Python 3.6.5 is the programming language that is employed in our work. Python is a generic purpose and programming language of high quality.

Technical Requirements	Training	Testing
Hardware	Xeon Processor	Intel i3 Processor
		5^{th} generation or above
	32 GB RAM	4GB RAM or above
		2GB Graphics
Software	Windows 10	Windows 10
	Visual Studio code	
	Cmd anaconda	Visual Studio code

TABLE I: TECHNICAL REQUIREMENTS

Technical requirements for hardware and software used in the training and testing phase has been depicted in the TABLE I.
The programming language Python3 was used for implementation work in conjunction with the libraries described as NumPy, Matplotlip, Scikit-learn, Pandas, SciPy and OpenCV. Keras version 2.0.8 is used as succeeding APIs for neural high-level network. Tensorflow version 1.4 is used. Platform used for training purpose is Google Cloud Platform (GCP). GCP used for processing is of high configuration that is Xeon processor, 32GB RAM and windows 10 operating system. Miniconda3, the version of Anaconda is used as data science platform and python IDE where we used both console and GUI.

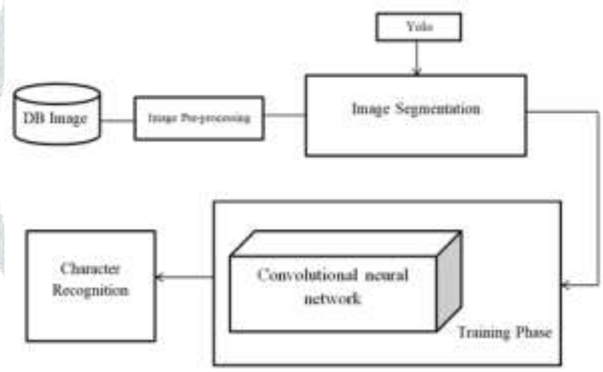

Fig. 4: Flowchart of proposed work

The proposed work flowchart is shown in Fig. 4. After the database image is fed for pre-processing the segmentation process is carried out by using YOLO in a darkflow framework. The segmented image is further moved in the training phase where epoch runs performing iteration. Approx 50 times epochs were performed in training phase which took more than 12 hours to complete. And, finally character recognition are done by using Convolution neural network. Fig. 5 displays some sample images from the database. The image archive includes photos of vehicle rear views, captured under various lighting conditions such as sunny, snowy, rainy, dusk, night-light.

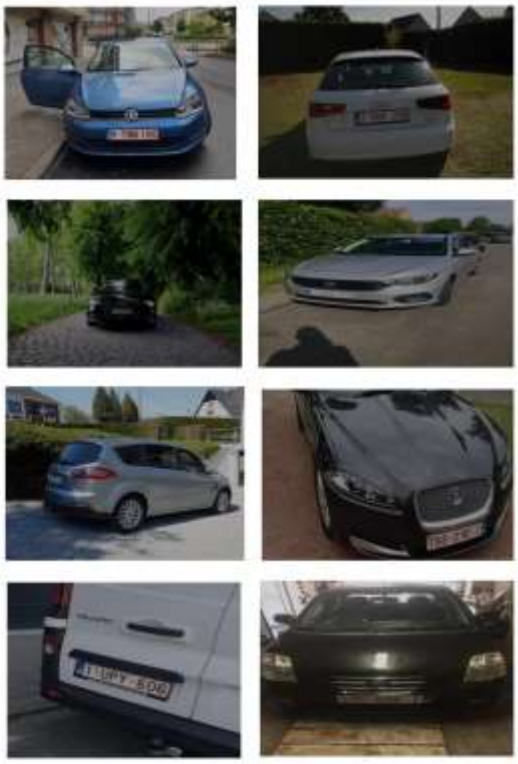

Fig. 5: Sample Images from Database

This data is then used for the second level model and the second level model makes the final prediction. In this model, Leaky Relu and softmax activation are used and an optimal learning rate is used. The hyperparameters during training of the architecture has been displayed in TABLE II below, showing that the values for the YOLO plate detection are saved in the cfg file format.

Testing Batch	1
Subdidvisions	1
Training Batch	64
Subdivisions	8
Width * Height	608×608
Channels	3
Momentum	0.9
Decay	0.0005
Angle	0
Saturation	1.5
Exposure	1.5
Hue	.1
Learning_rate	0.001
Burn_in	1000
Max_batches	500200
Policy	Steps
Steps	400000, 450000
Scales	.1, .1

TABLE II: Training Parameters used for Architecture

The output is performed at different scales by using anchor boxes for every item of the grid to perform object detection. The details of anchor boxes and the scales description along with the region details and the optimal values can be viewed in Table III and Table IV depicted below.

Region Description	
Anchors	0.57273, 0.677385, 1.87446, 2.06253, 3.33843, 5.47434, 7.88282, 3.52778, 9.77052, 9.16828
bias_match	1
Classes	1
Cords	4
Num	5
Softmax	1
Jitter	.3
Rescore	1

TABLE III: Region description in plate and character detection in YOLO

object_scale	5
noobject_scale	1
class_scale	1
coord_scale	1
Absolute	1
Thresh	.1
Random	1

TABLE IV: Plate detection scaling values and optimal values

In the following Fig. 6, the plot for the Accuracy Graph during Training, which reflects the percentage of correctly identified plates from the dataset can be seen.

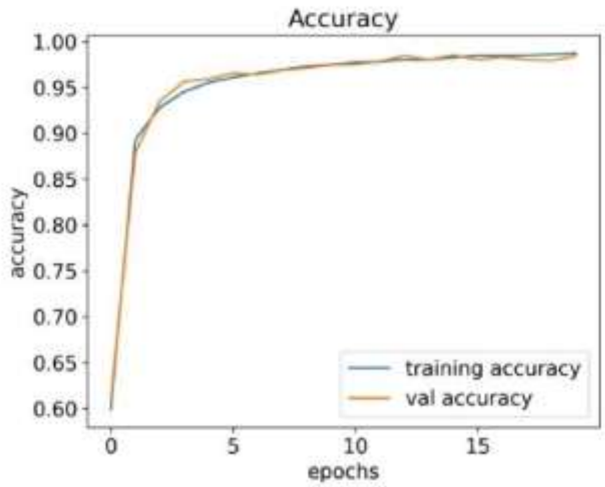

Fig. 6: Accuracy Graph during Training representing percentage of plates correctly recognized.

Deep learning aim is to reduce the gap between the output expected and the true output by finding the optimal weight value. This is also known as a Cost (C) or Loss function. The training loss plot in the test set can be viewed in Fig. 7 that depicts the gradual decrease in cost function.

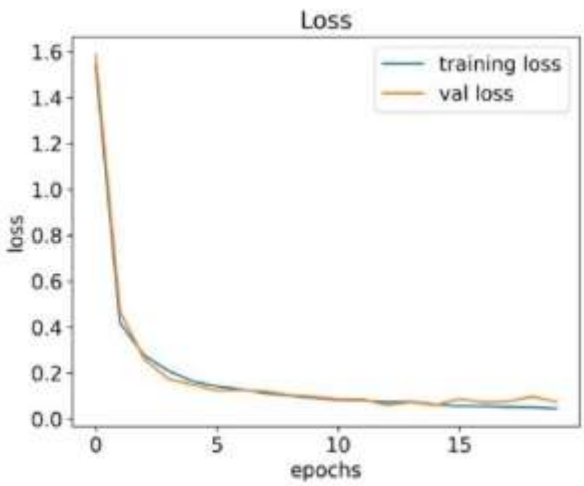

Fig. 7: Test Loss Graph during Training representing the gradual decrease in cost function.

We performed several iterations with different weights to achieve reduction in cost function. The optimizer used in this work is an Adam Optimizer. It aid in finding the minimum cost. The loss used is a sparse categorical cross entropy loss.

In the first cropping, the values of x and y are cut from an image in the form of x topleft, y topleft, x bottomright and y bottomright. The first crop image is not seen on the window as the processing is done on the back-end and it immediately disappears after the cropping. The flow of video processing in the main window and video functionality process has been shown in the Fig. 8.

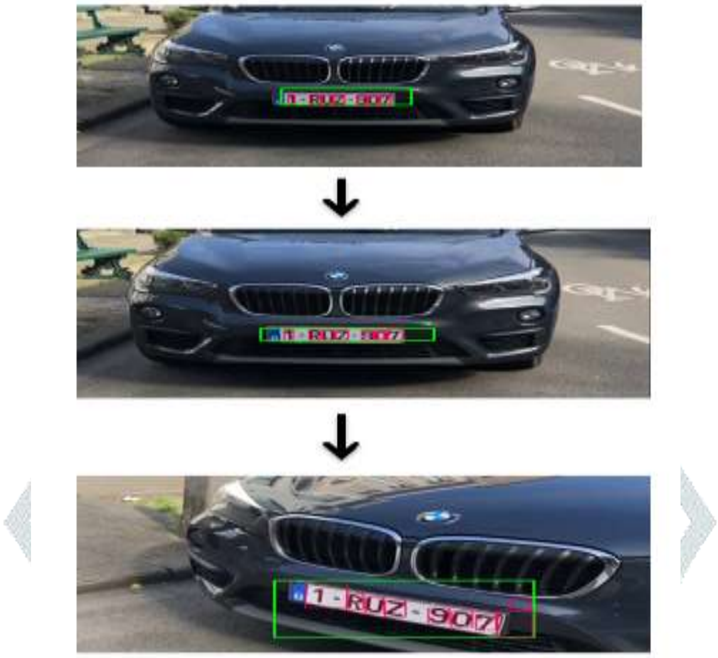

Fig. 8: Image showing the flow of video functionality process

In the second crop, the rectified image of plate is achieved as shown in Fig. 9. In this section the gray scale, threshold and contours are applied. The function cv2.canny returns an edged image. Figure 9 displays the pixel values as x=425, y=62 and the RGB values are R=43, G=44, B=49.

Fig. 9: Result for second crop plate window displaying the coordinate values.

OpenCV character segmented image is shown with its pixel value. The numbers 0-9 and alphabets A-Z are stored in dictionary variable. The image is reshaped and character predictions are made. Fig. 10 displays the pixel values as x=450, y=35 and the RGB values are R=80, G=86, B=107.

Fig. 10: Result for OpenCV character Segmentation window displaying the coordinate values.

In the YOLO character segmentation, the rectified image of plate is achieved as shown in Fig. 11, which displays the pixel values of x=170, y=114 and the RGB values are R=9, G=9, B=17.

Fig. 11: Result for YOLO character Segmentation window displaying the coordinate values.

VI. RESULTS AND DISCUSSION

This section offers accuracy based experimental results. The working of proposed model interprets the following results as shown in the Fig.12.

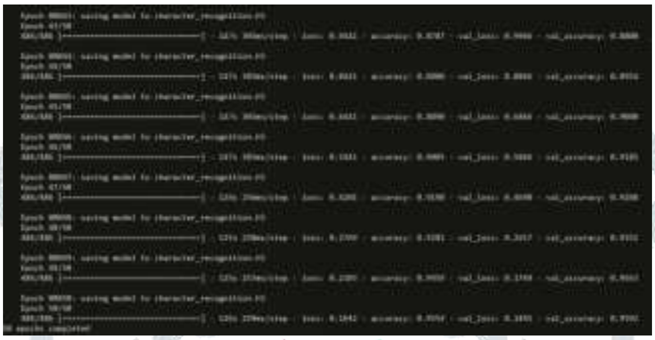

Fig. 12: Accuracy for LP detection in terms of training and validation of model

The training loss calculation and the training accuracy of the proposed model in license plate detection is 16.42 % and 95.54 % respectively, while the val_loss and val_accuracy measurements on the validation set are 14.91 % and 95.92% respectively. Since the validation accuracy is slightly more than the training accuracy by 0.0038 i.e. 0.38%, it depicts that the proposed model is under fitting the data. We need to avoid the over fitting of the model. It is the matrices on the validation set that measures the quality of model.

The comparison of license plate detection algorithms from previous researches is shown in TABLE V.

S. No.	Author	Method Used	Accuracy Rate
1	Saqib Rasheed *et. al* (2012)	Hough lines and Template Matching	94.11%
2	Sumanta Subhadhira *et. al* (2014)	Extreme Learning Machines (ELM)	89.05%
3	Yoshihiro Shima (2016)	CNN + SVM	89.7%
4	Zied Selmi *et. al* (2017)	Deep Learning CNN	93.80%
5	Cheng-Hung Lin *et. al* (2019)	Mask R-CNN	91%
6	Our Proposed Work	YOLO + CNN	95.92%

TABLE V: A Comparison of LP detection results by different approaches

The experimental results show that our proposed work, comprising of object detection algorithm YOLO and Deep neural network CNN, outperforms the other object detection methods with a high validation accuracy of 95.92%. It indicates that the method of detection of objects by YOLO is more effective than the conventional methods of detection. Fig. 13 shows graphical comparison of detection methods with various approaches.

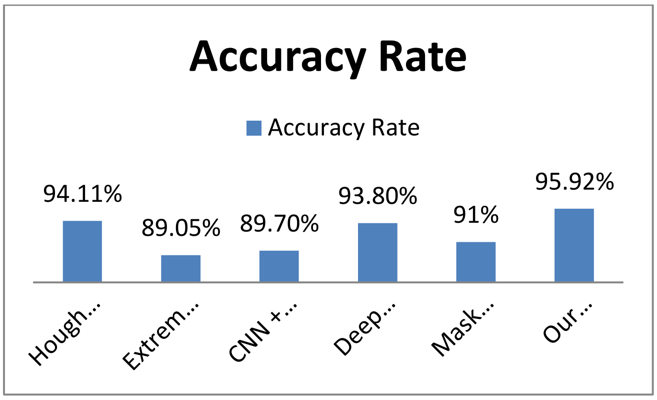

Fig. 13: Graphical Comparison of Detection Methods.

The experimental results show that our proposed work, comprising of object detection algorithm YOLO and Deep neural network CNN, outperforms the other object detection methods with a high validation accuracy of 95.92%. This shows that ensemble methods are more efficient than linear models and deep neural network. As shown, deep neural network efficiency is mainly due to the use of single neural network in object detection, greater flexibility and the ability to train efficiently on larger datasets. The proposed model performs substantially better than other previous models as it has a lower model complexity, it trains faster and has been optimized for effective use of memory and hardware resources, as well as for efficient training on large data sets.

VII. CONCLUSION

In this research, we used the You-Only-Look-Once YOLOv2 object detection technique for license plate detection and assessed their performance based on the custom dataset. Experimental results reveals that the measures of training loss and training accuracy of license plate detection of proposed model is 16.42% and 95.54% respectively Whereas the measures of val_loss and val_accuracy on the validation set is found to be 14.91% and 95.92% respectively. Therefore, it can be concluded that our proposed work which comprises of object detection algorithm YOLO and Deep neural network CNN, outperforms the other object detection methods with a high validation accuracy of 95.92%. It indicates that the method of detection of objects by YOLO is more effective than the conventional methods of detection. The proposed model performs significantly better than other previous models because it has a lower model complexity and it trains faster. This research opens up new possibilities in future recognition systems by providing the ability to use high-resolution cameras with an increased number of frames for higher accuracy and better recognition performance. There are various other standard number plate datasets where these techniques haven't been used yet. In future, this work can also be open for improvement, in regards to reading characters with higher accuracy rate. Here, for the purpose of validation, we used videos, but for future enhancements the code for images can be implemented as well. In this research, the design is successful in detecting single cars on highways, but for further improvements, detection of more than one vehicle at a time can enhance the present work.

REFERENCES

[1] Izidio, D. M., Ferreira, A. P., Medeiros, H. R., & Barros, E. N. D. S. (2019). An embedded automatic license plate recognition system using deep learning. Design Automation for Embedded Systems, 1-21.

[2] Li, H., Yang, R., & Chen, X. (2017, December). License plate detection using convolutional neural network. In 2017 3rd IEEE International Conference on Computer and Communications (ICCC) (pp. 1736-1740). IEEE.

[3] Laroca, R., Severo, E., Zanlorensi, L. A., Oliveira, L. S., Gonçalves, G. R., Schwartz, W. R., & Menotti, D. (2018, July). A robust real-time automatic license plate recognition based on the YOLO detector. In 2018 international joint conference on neural networks (ijcnn) (pp. 1-10). IEEE.

[4] Saif, N., Ahmmed, N., Pasha, S., Shahrin, M. S. K., Hasan, M. M., Islam, S., & Jameel, A. S. M. M. (2019, October). Automatic License Plate Recognition System for Bangla License Plates using Convolutional Neural Network. In TENCON 2019-2019 IEEE Region 10 Conference (TENCON) (pp. 925-930). IEEE.

[5] Rasheed, S., Naeem, A., & Ishaq, O. (2012, October). Automated number plate recognition using hough lines and template matching. In Proceedings of the World Congress on Engineering and Computer Science (Vol. 1, pp. 24-26).

[6] Subhadhira, S., Juithonglang, U., Sakulkoo, P., & Horata, P. (2014, March). License plate recognition application using extreme learning machines. In 2014 Third ICT International Student Project Conference (ICT-ISPC) (pp. 103-106). IEEE.

[7] Liu, W. C., & Lin, C. H. (2017, May). A hierarchical license plate recognition system using supervised K-means and Support Vector Machine. In 2017 International Conference on Applied System Innovation (ICASI) (pp. 1622-1625). IEEE.

[8] Selmi, Z., Halima, M. B., & Alimi, A. M. (2017, November). Deep learning system for automatic license plate detection and recognition. In 2017 14th IAPR international conference on document analysis and recognition (ICDAR) (Vol. 1, pp. 1132-1138). IEEE.

[9] Lin, C. H., & Li, Y. (2019, August). A license plate recognition system for severe tilt angles using mask R-CNN. In 2019 International Conference on Advanced Mechatronic Systems (ICAMechS) (pp. 229-234). IEEE.

[10] Suvarnam, B., & Ch, V. S. (2019, March). Combination of CNN-GRU Model to Recognize Characters of a License Plate number without Segmentation. In 2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS) (pp. 317-322). IEEE.

[11] Shima, Y. (2016, December). Extraction of number plate images based on image category classification using deep learning. In 2016 IEEE International Symposium on Robotics and Intelligent Sensors (IRIS) (pp. 19-26). IEEE.

[12] Bhardwaj, D., & Kaur, H. (2014, October). Comparison of ML algorithms for identification of Automated Number Plate Recognition. In Proceedings of 3rd International Conference on Reliability, Infocom Technologies and Optimization (pp. 1-6). IEEE.

[13] Mondal, M., Mondal, P., Saha, N., & Chattopadhyay, P. (2017, December). Automatic number plate recognition using CNN based self synthesized feature learning. In 2017 IEEE Calcutta Conference (CALCON) (pp. 378-381). IEEE.

[14] Lee, S., Son, K., Kim, H., & Park, J. (2017, July). Car plate recognition based on CNN using embedded system with GPU. In 2017 10th International Conference on Human System Interactions (HSI) (pp. 239-241). IEEE.

[15] Zoran Kalafatić, Zadnja promjena, 2003, Information Technology Application Project Ministry of Science and Technology, viewed 01 January 2003 http://www.zemris.fer.hr/projects/LicensePlates/english/results.shtml

[16] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 779-788).

[17] Redmon, J., & Farhadi, A. (2017). YOLO9000: better, faster, stronger. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 7263-7271).

[18] Redmon, J., & Farhadi, A. (2018). Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767.

[19] LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 2278-2324.

[20] Zhao, Z., Yang, S., & Ma, X. (2008, December). Chinese license plate recognition using a convolutional neural network. In 2008 IEEE Pacific-Asia Workshop on Computational Intelligence and Industrial Application (Vol. 1, pp. 27-30). IEEE.

[21] Duffner, S. (2008). Face image analysis with convolutional neural networks (Doctoral dissertation, University of Freiburg).

[22] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems (pp. 1097-1105).