An Evaluation of Various Pre-trained Optical Character Recognition Models for Complex License Plates

Haziq Idrose[1], Nouar AlDahoul[1](✉), Hezerul Abdul Karim[1](✉), Rehan Shahid[2], and Manish Kumar Mishra[2]

[1] Faculty of Engineering, Multimedia University, Cyberjaya, Selangor, Malaysia
nouar.aldahoul@live.iium.edu.my, hezerul@mmu.edu.my
[2] Tapway Sdn Bhd, Petaling Jaya, Selangor, Malaysia

Abstract. Optical Character Recognition (OCR) has been investigated widely to recognize characters in images for various applications including license plate recognition. Several limitations and distortions are available in images such as noise, blurring, and closed characters (alphabet and numbers) which makes the task of recognition more complex. This paper addresses the closed characters and blurring problem utilizing three pre-trained deep learning OCR models including Pytesseract, EasyOCR and KerasOCR. We evaluated and compared these methods using a dataset that contains Malaysian license plates. The results show that KerasOCR was able to outperform other methods in terms of recognition accuracy. KerasOCR was able to recognize 107 images out of 264 images compared to only 87 images in EasyOCR and 97 images in Pytesseract.

Keywords: Optical Character Recognition · License Plate Recognition · Pre-trained deep learning models · KerasOCR

1 Introduction

License Plate Recognition is a technology found to automatically recognize characters from video footage. It is useful for various applications such as security and traffic analysis, vehicle monitoring in highways, and car parking in commercial buildings. OCR works by converting pictures that have characters to digital texts.

Several research works have demonstrated the application of car plate recognition with various methods that each has its advantages and disadvantages. Traditional image processing algorithms were utilized for automatic car-plate detection and recognition. The algorithm that was running on the Field Programmable Gate Array (FPGA) focused on recognition of numbers that have various shapes, sizes, fonts and colours in Malaysian license plates [1]. They used edge detection method to detect the license plate and OCR with correlation approach to recognize characters [1]. On the other hand, deep learning approaches have been demonstrated for characters recognition in car plates. License Plate Recognition via Deep Neural Network (LPRNet) is an End-to-End method without preliminary character segmentation. This method was trained using the lightweight convolution Neural Network (CNN) for real-time recognition [2]. Convolution Neural Networks based method was also able to achieve a high accuracy using small training dataset even if distortions and illumination are available in the images [3]. An evaluation of license plate recognition methods under various conditions such as low illumination and weather change (rain and snow) was also explored using various texture-based descriptors such as Histogram of Oriented Gradients (HOG) and Haar-Like Features with numerous classifiers such as Support Vector Machine (SVM) [4]. The complexity of environments including low-quality camera, adverse atmospheric conditions, unclear car plates, complex backgrounds, and uneven illumination can affect the recognition performance largely [5]. To overcome previous problems, End-to-End deep learning architecture was demonstrated. It includes a residual error network to extract basic features, a multi-scale net to extract multi-scale features, a regression net to locate plate and characters, and a classification net to recognize the characters [5]. The CNN-RNN based method for license plate recognition was found to include Convolutional Neural Networks (CNN) to extract features and Recurrent Neural Networks (RNN) such as B-LSTM (Bi-Directional Long Short-Term Memory), to extract context information based on the past information [6]. This combination of CNN and RNN was found to handle the poor quality, blur, noise, and complex background. YOLO object detector was utilized for license plate detection and layout classifications [7]. It showed robustness regardless of camera distance, lighting condition, and vehicle types. Generative Adversarial Network (GAN) was used to generate new synthetic images using a small set of real images [8]. A modified lightweight YOLOv2 model was used for End-To-End license plate character detection and recognition without segmentation [8]. Additionally, a super-resolution generative adversarial network (SRGAN) model was explored to overcome the low recognition rate of the CCTV [9]. They applied distortion-correction algorithm for the misrecognized characters to enhance the recognition rate [9]. They used YOLO2 model for character recognition.

Several OCR models that were trained with large datasets for text extraction and optical character recognition have been discussed in the literature. Three famous OCR models namely Pytesseract [10], EasyOCR [11], and KerasOCR [12] were utilized in this paper as pre-trained OCR models to evaluate their performance to recognize characters in car plates of our challenging dataset.

2 Materials and Methods

This section describes the dataset and discusses the methods utilized in this work.

2.1 Dataset Overview

The dataset used in this work consists of 264 blurry and closed characters images of Malaysian license plate. These images were considered complex and difficult to be recognized by state-of-the-art OCR methods. The pictures were collected by Tapway Sdn Bhd, Malaysia [13]. Figure 1 shows several samples from our dataset.

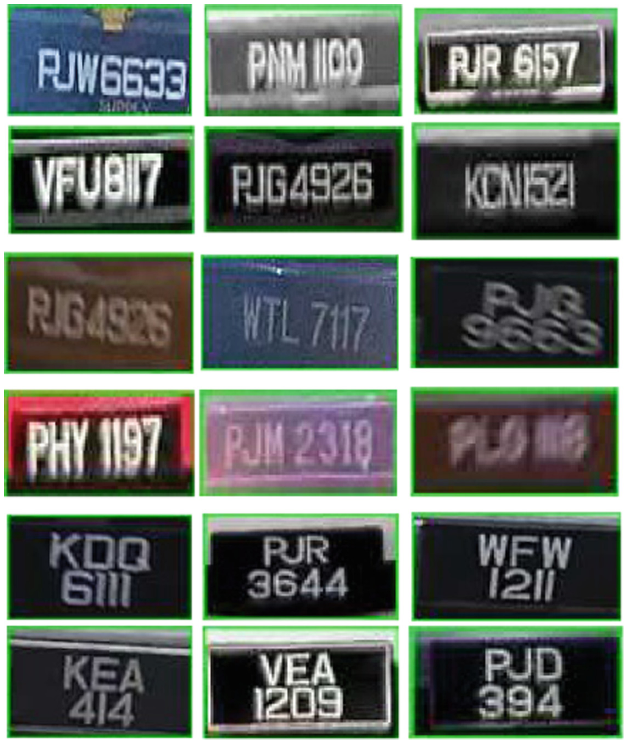

Fig. 1. Several samples from our dataset showing closed characters in rows (1 and 2), blurring in rows (3 and 4), and two-line characters in rows (5 and 6).

2.2 Pre-trained OCR Models

2.2.1 PyTesseract [10]

Tesseract can be utilized as an API to extract printed content from pictures. It supports several languages. Tesseract 4.0 added a new OCR engine based on Long Term Short Memory (LSTM) neural networks for OCR engine which is focused on line-recognition.

2.2.2 EasyOCR [11]

Text detection uses the Character-Region Awareness for Text detection (CRAFT) algorithm. The recognition model is a Convolutional Recurrent Neural Network (CRNN). It includes three components: feature extraction (Resnet) and Visual Geometry Group (VGG), sequence labelling (LSTM) and decoding (Connectionist Temporal Classification (CTC)).

2.2.3 KerasOCR [12]

KerasOCR uses CRAFT to detect text area by exploring each character region and affinity between characters. To find bounding box of texts, minimum bounding rectangles were found on binary map after thresholding scores of character region and affinity. For text recognition, original CRNN model, or spatial transformer network layer is used to rectify the text.

Several morphological operations were added to the original images before being applied to the input of pre-trained OCR models as an attempt to enhance the recognition performance.

1) An erosion was applied to an image to erode the foreground object and make it smaller.
2) A dilation was applied to an image to fill the holes in the object.
3) An opening was applied to an image by performing erosion followed by a dilation.
4) A closing was applied to an image by performing dilation followed by an erosion.

3 Results and Discussion

This section discusses the results of evaluating and comparing OCR models for car plate recognition task presented in this paper.

3.1 Experimental Setup

The experiments of evaluating OCR pre-trained models were carried out on Google Colab with K80 GPU. The algorithms were implemented using Python programming language with Tensorflow and Torch frameworks.

3.2 Experimental Results

Figure 2 shows several samples that were predicted correctly by the three OCR pre-trained models used. The three models have several problems and limitation in car plate recognition task which can be summarized as follow:

1) When the image of license plates has two lines of characters, some models cannot recognize text well.
2) Characters are confused with each other such as '1'was confused with 'I', 8' was confused with 'B', and 'Q' and 'O' were also confused.
3) When the picture is blurred, all OCR models were not able to recognize characters well.
4) When the characters are too closed, the performance of OCR models also degraded.

While EasyOCR and Pytesseract suffer from previous limitations such as two text lines and character confusion, these limitations have been addressed by Keras-OCR which was found to outperform other OCR models in terms of accuracy (number of correct recognized car plates) as shown in Table 1. Figure 2 shows several samples that were predicted correctly by the three OCR pre-trained models used. Figure 3 shows several samples that were predicted correctly by only KerasOCR.

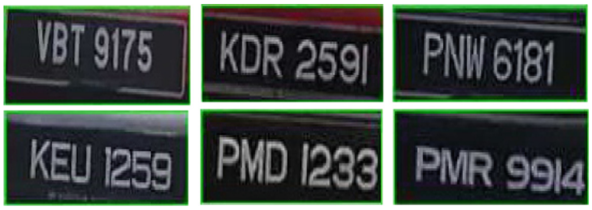

Fig. 2. Several samples that were predicted correctly by all models

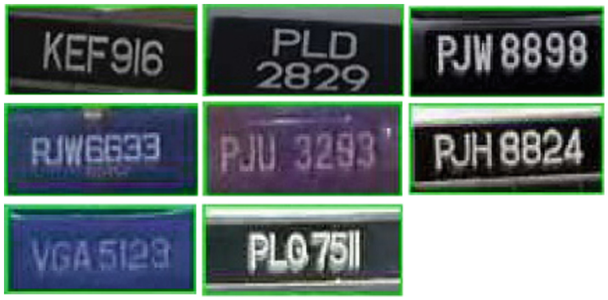

Fig. 3. Several samples that were predicted correctly by only KerasOCR

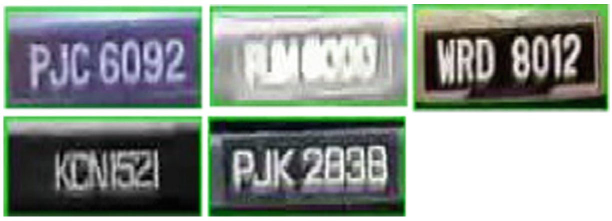

Fig. 4. Several samples that were predicted wrongly by all models

However, there are still several pictures of car plates that cannot be recognized even by KerasOCR as shown in Fig. 4 because they are so blurry and have so closed characters. Therefore, retraining KerasOCR with this dataset of car plates is one of the potential solutions to address the previous limitations.

Table1 presents the recognition accuracy by calculating number of correctly recognized car plates over all plates. The accuracy of pre-trained KerasOCR was 40.53% which is higher than Pytesseract by about 4% and higher than EasyOCR by about 7.5%. In summary, Tesseract can perform well for high-resolution images. Morphological operations cannot help to increase performance of Tesseract. On the other hand, EasyOCR with opening or closing morphological operations produced better results than using original pictures. Finally, Keras-OCR gave good results even if there was a slight blur in images and the characters were somehow closed and the characters were spread in two lines. Morphological operations cannot help to increase performance of Keras-OCR.

Table 1. The comparison between the three pre-trained models in terms of recognition accuracy

Methods	Pytesseract	EasyOCR	KerasOCR
original	97	79	**107**
Dilation	73	77	82
Erosion	72	79	87
Opening	73	87	86
Closing	72	87	82
Accuracy %	36.74	32.95	**40.53**

4 Conclusion and Future Work

This paper demonstrated the task of license plate recognition with blurring and closed characters (alphabet and numbers). Because of the lack of samples, three OCR pre-trained models including Pytesseract, EasyOCR and KerasOCR were utilized. A dataset of 264 pictures of Malaysian car plates was used for evaluation and comparison. The experimental results showed that KerasOCR method outperformed other models in terms of recognition accuracy. It was found that 107 images out of 264 images of license plates were recognized correctly using KerasOCR compared to only 87 images in EasyOCR and 97 images in Pytesseract.

The limitation of this work is that several images were still not recognized correctly by pre-trained KerasOCR. Therefore, training KerasOCR utilizing license plates with closed characters is the future work. This task required collection or synthetic generation of large number of license plate pictures with closed characters [14]. Additionally, digit segmentation methods have a big potential to improve detection and recognition of characters using state-of-the-art object detection methods.

Acknowledgments. This research project was funded by Multimedia University, Malaysia.

References

1. Ng Simin, Florence Choong Chiao Mei, "Automatic Car-plate Detection and Recognition System", 2013.
2. Sergey Zherzdev, Alexey Gruzdev, 2018, "LPRNet: License Plate Recognition via Deep Neural Networks", r arXiv:1806.10447v1, https://doi.org/10.48550/arXiv.1806.10447
3. Madhusree Mondal, Parmita Mondal, Nilendu Saha and Paramita Chattopadhyay, "Automatic number plate recognition using CNN based self synthesized feature learning," 2017 IEEE Calcutta Conference (CALCON), 2017, pp. 378-381, doi: https://doi.org/10.1109/CALCON.2017.8280759.
4. Rio-Alvarez, J. de Andres-Suarez, M. Gonzalez-Rodriguez, D. Fernandez-Lanvin, and B. López Pérez, "Effects of Challenging Weather and Illumination on Learning-Based License Plate Detection in NonControlled Environments", 27 Jun 2019, Scientific Programming, V 2019,https://doi.org/10.1155/2019/6897345

5. Di Wang, Yumin Tian, Wenhui Geng, Lin Zhao, Chen Gong, February 2020, "LPR-Net: Recognizing Chinese license plate in complex environments",Pattern Recognition Letters, V 130, pp 148-156, https://doi.org/10.1016/j.patrec.2018.09.026
6. Palaiahnakote Shivakumara1, Dongqi Tang, Maryam Asadzadehkaljahi1, Tong Lu, Umapada Pal, Mohammad Hossein Anisi, "The CNN-RNN based method for license plate recognition", 2018, CAAI Transactions on Intelligence Technology, pp 1-8, https://doi.org/10.1049/trit.2018.1015
7. Rayson Laroca, Luiz A. Zanlorensi, Gabriel R. Goncalves, Eduardo Todt1, William Robson Schwartz, David Menotti1, 9 March 2021, "An Efficient and Layout-Independent Automatic License Plate Recognition System Based on the YOLO detector", IET Intelligent Transport Systems, vol. 15, no. 4, pp. 483-503, 2021. https://doi.org/10.1049/itr2.12030
8. Byung-Gil Han, Jong Taek Lee, Kil-Taek Lim and Doo-Hyun Choi, 16 April 2020, "License Plate Image Generation using Generative Adversarial Networks for End-To-End License Plate Character Recognition from a Small Set of Real Images", Applied Science, 10(8), 2780, pp 1-16, https://doi.org/10.3390/app10082780
9. Tae-Gu Kim, Byoung-Ju Yun, Tae-Hun Kim, Jae-Young Lee, Kil-Houm Park, Yoosoo Jeong, and Hyun Deok Kim 1, 2021, "Recognition of Vehicle License Plates Based on Image Processing", Applied Science, 11(14), 6292, pp 1-12 https://doi.org/10.3390/app11146292
10. Tesseract documentation, April. 2022, [online] Available: https://tesseract-ocr.github.io/
11. EasyOCR, April. 2022, [online] Available: https://github.com/JaidedAI/EasyOCR
12. keras-ocr, April. 2022, [online] Available: https://github.com/faustomorales/keras-ocr
13. VehicleTrack, April. 2022, [online] Available: https://gotapway.com/solutions/vehicletrack
14. Text Recognition Data Generator, April. 2022, [online] Available: https://github.com/Belval/TextRecognitionDataGenerator