An Application Combining CNN-BLSTM with CTC for License Plate Recognition

Pham Tuan Dat

December 19, 2021

An application combining CNN-BLSTM with CTC for license plate recognition

Pham Tuan Dat
Faculty of Information Technology
Maritime University
Hai Phong, Vietnam
datpt@vimaru.edu.vn

Abstract—**This paper proposes an application based on convolutional neural network-bidirectional long short-term memory and connectionist temporal classification for the problem of license plate recognition. Unlike previous machine learning models, the hybrid network handles variable-length sequences and does not need to segment characters from plates. The application experiments with a model of hybrid network, evaluates its correct predictions on the validation set by BLEU scores, and compares it with the model of K-nearest neighbors and the model of Support vector machines. Estimated BLEU scores show that the model of hybrid network gives a reliable accuracy, and this result is better than that of two machine learning models.**

Keywords: convolutional; recurrent; connectionist; beam

I. INTRODUCTION

The number of vehicles in cities has grown quickly and it is not easy to control security and traffic without an efficient solution to the problem of license plate recognition. For instance, if the speed of a car on the road is higher than the maximal speed, law enforcement agencies need to recognize the plate of that car before finding out the owner's identity.

License plate recognition is not a new subject and there are previous approaches to this problem. Typical models of machine learning are K–nearest neighbors (KNN) and Support vector machines (SVM) [10,11], which give really impressive results. However, a difficulty of these models is that they must divide each plate into discrete patterns before performing the learning stage on patterns. If models lack effective image processing operations, then they obtain poor recognition results.

This paper proposes an application that combines the advantages of convolutional neural network (CNN) and bidirectional long short-term memory (BLSTM). Furthermore, the application applies Connectionist Temporal Classification (CTC) to the hybrid network. CTC does not require specific information of the input data and it allows the hybrid network to train the input data without segmentation phases [6]. In other words, the CNN-BLSTM can overcome the drawbacks of the machine learning approach.

The application experiments with the CNN-BLSTM on the training and validation sets. The application evaluates the accuracy of this hybrid network on the validation set by BLEU scores [13] and compares it with SVM and KNN models. In fact, the hybrid network gives a noteworthy improvement in training irregular plates or poor quality plates. Experiments show that the hybrid network can achieve average BLEU score from 0.86 to 0.92 on the validation set. Meanwhile, both the SVM and KNN models obtain maximal BLEU score no more than 0.9.

II. BACKGROUND

A. *Convolutional Neural Network and Bidirectional Recurrent Neural Network*

In recent years, CNN has been widely applied in deep learning fields such as object recognition and computer vision. Along with improvements in optimization algorithms, the published results of projects indicated that CNN could achieve high performance for those subjects.

The structure of CNN [1] is quite different from that of a multi-layer perceptron network. Except for input and output layers, CNN consists of three types of layers and each one handles a specific function. Convolutional layers play a crucial role in CNN when they use kernel filters to extract local features of images. Many various features can increase the recognition accuracy of CNN. Due to a large number of parameters for feature maps, so pooling layers reduce this burden by down-sampling operations. Lastly, feature maps of pooling layers or convolutional layers are connected to one fully connected layer, which takes all neurons in the previous layer and connects them to every single neuron of the current layer to generate global semantic information.

CNN model takes the input data as a set of discrete patterns. If a CNN model faces sequential data, it does not train sequences directly. In this case, each sequence must be split into discrete patterns before each pattern can be learned by that model.

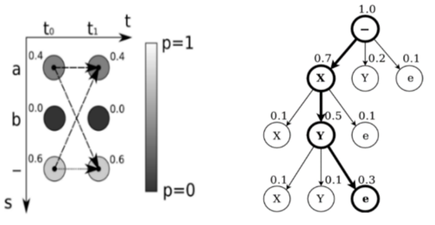
Recurrent neural network (RNN) is an appropriate complement to CNN. RNN provides a powerful mechanism for handling variable-length sequences. Ordinary RNN includes the input layer, output layer, and hidden layer with recurrent connections. Although ordinary RNN has difficulty in carrying information over a long distance, its several variants are able to deal with this limitation. LSTM is such an example, it does not just inherit common features of ordinary RNN but also controls better flows of information between neurons than ordinary RNN. A unit in the hidden layer of LSTM is a memory block containing at least one memory cell and three gates. "Input gate" allows LSTM to keep or override information in a memory cell, LSTM uses "Output gate" to decide when to access information in a memory cell and when to prevent a memory cell from perturbing the remainder of a network, "Adaptive forget gate" learns to reset the content of a memory cell if the information is no longer necessary [2,3]. As a result, LSTM can avoid exploding or vanishing in a large number of time steps.

RNNs only use past context while information from both forward-backward directions is useful and complementary to each other. Schuster and Paliwal propose a bidirectional recurrent neural network [4] (BRNN) including two layers, the first layer is in charge of the forward direction and the second one for the backward direction. The output from forward states is not connected to the input of backward states, and vice versa. Future and past information of the currently evaluated time frame can directly be used to minimize the objective function without the need for delays.

B. Connectionist Temporal Classification

Several hybrid networks such as RNNs with hidden Markov model (HMM) are studied to cope with sequence labeling. Nevertheless, HMM requires a state model design and needs dependency assumptions for making inferences. Moreover, these hybrid models do not show the full potential of RNNs for sequence modeling.

CTC is an alternative to HMM, when decoding with this mechanism, RNNs do not need specific knowledge of the input data, and they do not also have to perform the segmentation phase. The principle of CTC is that it interprets the output of RNNs into a probability distribution over label sequences [6].



(a) (b)

Figure 1. (a) An example of CTC Decoding; (b) PS decoding on the label alphabet X,Y.

From this probability distribution, if the objective function in (2) is minimized, CTC can maximize the probabilities of correct label sequences. To achieve this, RNN models also need to train the input data with an optimizer.

$$\begin{cases} h(x) = \arg \max\limits_{l \in L^{\leq T}} p(l \mid x) \\ p(l \mid x) = \sum\limits_{\pi \in B^{-1}(l)} p(\pi \mid x) \end{cases} \quad (1)$$

$$O(S, N_w) = -\sum_{(x,z) \in S} \ln(p(z \mid x)) \quad (2)$$

$$\begin{cases} h(x) \approx B(\arg \max\limits_{\pi \in N^t} p(\pi \mid x)) \\ p(\pi \mid x) = \prod\limits_{t=1}^{T} y_{\pi_t}^t, \forall \pi \in L^{'T} \end{cases} \quad (3)$$

There are some approximate decoding methods, which give relatively good results in practice. The first one is called "Best path decoding" (BP), which assumes that the most probable path will correspond to the most probable labeling, as defined in (3). Nonetheless, in several situations, this decoding mechanism does not predict results as expected. In Fig. 1a, BP algorithm generates the labeling "" (probability = 0.6*0.6 = 0.36) but the correct answer is "a". By increasing the number of candidates (best beams) at each time-step, the sum of the probabilities of possible paths yielding 'a': 2*0.6*0.4 + 0.4*0.4 = 0.64. Therefore, "Vanilla beam search decoding" (VBS) [7] may improve the accuracy of decoding in such situations.

"Prefix search decoding" (PS) divides the output sequence into sections that are very likely to begin and end with a blank. CTC chooses boundary positions where probabilities of blank labels are above a given threshold. In Fig.1b, the number above an end node ('e') is the probability of the single labeling ending at its parent. The search ends when the labeling "XY" is more probable than any remaining prefix.

III. RELATED WORK

A. Functions in the Application

The experimental application includes three main functions in Fig. 2a: Pre-processing and Extraction, CNN-BLSTM, and CTC Decoder.

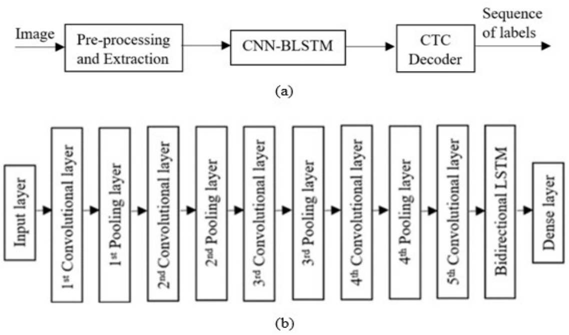


Figure 2. (a) Functions of the application; (b) Structure of the CNN-BLSTM.

In the first function, each image in the data set should be pre-processed before the detection phase. Initially, the image is converted into a gray image. Photos of the data set are taken in actual conditions, so the application adjusts the contrast of each image, eliminates noise, and enhances borders in images with Blur and Sobel filters. Next, the application transforms gray images into binary images by Otsu algorithm [12].

To extract plates in images, the idea is that the application detects contours bounding plates. The popular approach is an algorithm defining relations between the borders on a binary image [8]. In this case, plates have the shape of a rectangle so each contour is estimated according to the proportion of the width to the height. However, after the pre-processing phase, the application may not detect contours bounding regions yet, so morphological processing operations [9] on images might increase the probability of finding out contours.

The direction of each plate often is deviated an angle from the horizontal axis. The transformation approximately calculates each deviation angle from vertex coordinates. Obviously, this performance is only good for small deviation angles. Otherwise, the transformation makes images distorted and does not improve the effectiveness of recognition.

As shown in Fig. 3, some plates in the first two lines have irregular shapes. The application improves quality and restores these plates to regular shapes as illustrated in the remaining lines.

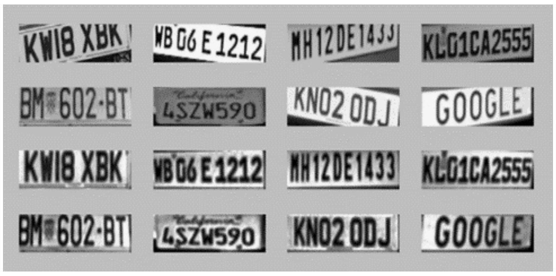


Figure 3. Plates are pre-processed and extracted from images.

The next function is the CNN-BLSTM. In Fig. 2.b, the convolutional and pooling layers in the network are placed alternatively. The network sets up the number of filters from 64 to 256 for convolutional layers. Pooling layers use Max function while the activation in the layers is Relu [1].

The network installs a bidirectional LSTM instead of an ordinary RNN. Each LSTM consists of 128 units, it uses sigmoid and hyperbolic tangent activations [14] on gates and cell states. In the LSTMs, these activations accelerate faster convergence than Relu in learning phases.

The dense layer in the CNN-BLSTM does not predict the final output. It provides a matrix containing character probabilities for the CTC Decoder (the last function). From softmax probabilities and given labels, the Decoder will predict the output sequence. In this case, the application supports both BP and VBS algorithms for the CTC layer.

Both Rmsprop and Adam optimizers make the network converge quickly but Rmsprop lacks stability during the learning phases. Thus, the application chooses Adam [5] for the network. Besides, there are other essential functions inserted into the hybrid network to accelerate convergence.

B. Experiment and Comparison

The application uses programming languages and relevant libraries such as C++, Python, and OpenCV. The small data set (904 samples) is collected from addresses: “www.kaggle.com/andrewmvd/car-plate-detection”, “www.kaggle.com/pcmill/license-plates-on-vehicles”, and some other sources on the Internet. This set is split into the training set and the validation set in the proportion 7:3. The validation set is used for tuning hyper-parameters in training stages but it is also used to evaluate the accuracy of the network. The application shuffles samples of the data set, so the network can experiment on different samples of the training and validation sets in each session.

Fig. 4 illustrates the losses of the CNN-BLSTM on two sets in a training phase. The common figure of training phases is that the estimated losses decline fast in the first 10 epochs. Afterward, these losses have dropped very slowly. The CNN-BLSTM gains a very low loss on the training set and a high loss on the validation set. One reason for this outcome is that

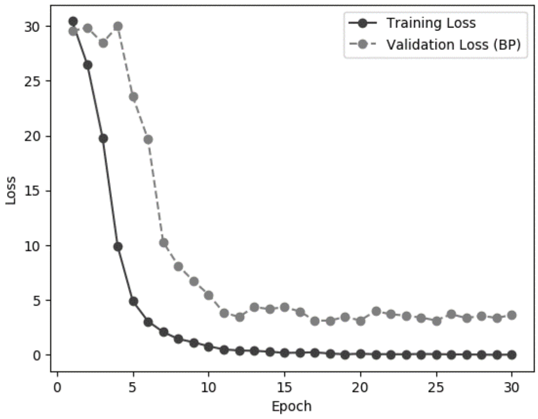


Figure 4. Estimated losses in a training phase.

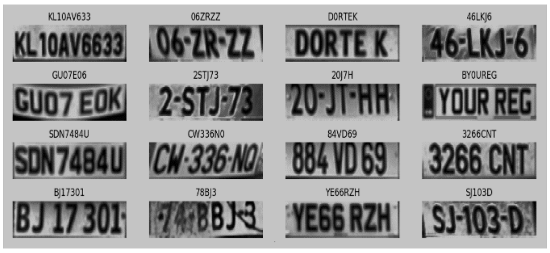


Figure 5. Several plates in the validation set are decoded by VBS.

the training data does not have a large number of samples. Although there is a remarkable difference between the losses on two sets, BLEU score on the validation set is also good, around 0.92. In other words, the network gets a reliable accuracy.

Table I shows that both BP and VBS algorithms allow the CNN-BLSTM to generate highly accurate predictions on the validation set. On the theoretical basis, VBS may cope with the weakness of BP, but on this data set, BP gives an overall performance better than VBS.

Of course, the CNN-BLSTM still confuses characters 'I' with '1', 'S' with '5', 'G' with '6', 'M' with 'W', etc. Additionally, many plates contain symbols or information about regions, so it mistakes such information and extracts incorrect texts (Fig. 5). But if working on a large data set, the CNN-BLSTM makes fewer mistakes.

The CNN-BLSTM is not the first model to solve the problem of license plate recognition. To have an objective assessment of the accuracy of the CNN-BLSTM, the application compares it with the SVM and KNN models on the same data set. The SVM and KNN take and train a set of discrete patterns, so the application must segment samples in the data set into discrete patterns. There are 2778 patterns in classes from'A' to 'Z' and '0' to '9' for the training phases.

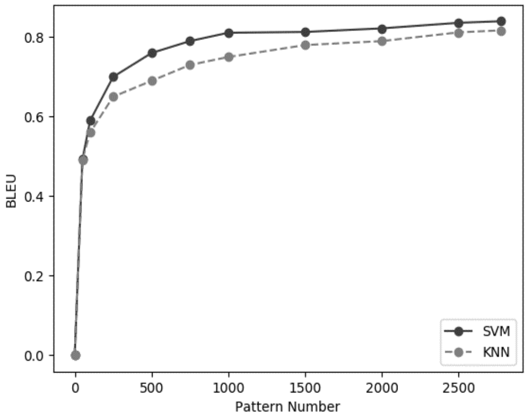


Figure 6. BLEU scores of the SVM and the KNN.

In Fig.6, if the number of patterns in the training set is small, BLEU scores (on a test set) of the SVM and the KNN increase quickly. Afterward, when increasing the number of patterns over 2000, then two models have insignificant improvements in terms of accuracy. BLEU scores of two models on the test set (validation set of the hybrid network) are lower than 0.9.

The SVM exactly predicts most samples of the test set but sometimes it is not able to train irregular plates. The result of segmentation much depends on image processing operations. Ineffective operations make conventional models difficult to train and recognize such plates. Meanwhile, the KNN only correctly predicts patterns from images, which are taken in ideal conditions, so it gives the worse accuracy with BLEU score below 0.85 (Table II). On the contrary, the CNN-BLSTM trains almost all samples, so in the case of recognizing one or more characters incorrectly for each plate, then its BLEU score still exceeds that of the SVM and KNN models.

TABLE I. BLEU SCORES ON THE VALIDATION SET

Algorithm	**Cumulative 1-Gram**	**Cumulative 2-Gram**
BP	0.92	0.88
VBS	0.89	0.86

TABLE II. BLEU SCORES OF MODELS

Model	**Cumulative 1-Gram**	**Cumulative 2-Gram**
CNN-BLSTM	0.92	0.86
SVM	0.89	0.85
KNN	0.84	0.78

IV. CONCLUSION

The approach of machine learning obtains very impressive results for the problem of license plate recognition. Nonetheless, a difficulty of these models is that they must segment plates into patterns before training phases. A hybrid network taking the advantages of convolutional neural network and recurrent neural network is another approach, which can train a data set including irregular plates or poor quality plates.

In this paper, the application builds a hybrid network of CNN-BLSTM and a Decoder of CTC for the problem. The application evaluates the accuracy of this hybrid network on the validation set by BLEU scores (from 0.86 to 0.92). These results are better than BLEU scores of KNN and SVM models.

REFERENCES

[1] Jiuxiang Gu, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, Li Wang, Gang Wang, Jianfei Cai, and Tsuhan Chen, "Recent advances in convolutional neural networks", Elsevier, October 2017.

[2] Sepp Hochreiter and Jurgen Schmidhuber, "Long Short-Term Memory", Neural Computation, 1997.

[3] Felix A.Gers, Jurgen Schmidhuber, and Fred Cummins, "Learning to Forget: Continual Prediction with LSTM", Neural Computation, 2000.

[4] Mike Schuster and Kuldip K. Paliwal, "Bidirectional Recurrent Neural Networks", Transactions on Signal Processing, vol. 45, no. 11, November 1997.

[5] Diederik P.Kingma and Jimmy LeiBa, "Adam: a Method for Stochastic optimazation", ICLR, 2015.

[6] Alex Graves, Santiago Fernandez, Faustino Gomez, and Jurgen Schmidhuber, "Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks", Proceedings of 23rd International Conference on Machine Learning, 2006.

[7] Harald Scheidl, Stefan Fiel, and Robert Sablatnig, "Word Beam Search: A Connectionist Temporal Classification Decoding Algorithm", 16th International Conference on Frontiers in Handwriting Recognition, 2018.

[8] Satoshi Suzuki and Keiichi Abe, "Topological Structural Analysis of Digitized Binary Images by Border Following", Computer Vision, Graphics, and Image Processing, pp.32-46, 1985.

[9] Pinaki Pratim Acharjya and Esha Dutta, "Image Segmentation and Contour Recognition Based on Mathematical Morphology", International Journal of Computer and Information Engineering, vol. 11, 2017.

[10] D Gunawan, W Rohimah, and R F Rahmat, "Automatic Number Plate Recognition for Indonesian License Plate by Using K-Nearest Neighbor Algorithm", IOP Conference Series: Materials Science and Engineering, 2019.

[11] Sahar S. Tabrizi and Nadire Cavusa, "A hybrid KNN-SVM model for Iranian license plate recognition", 12th International Conference on Application of Fuzzy Systems and Soft Computing, 29-30 August 2016.

[12] Nobuyuki Otsu, "A Threshold Selection Method from Gray-Level Histograms", IEEE Transactions on Systems, man, and cybernetics, vol. smc-9, no. 1, January 1979.

[13] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu, "BLEU: A Method for Automatic Evaluation of Machine Translation", Proceedings of ACL, pp. 311-318, 2002.

[14] Chigozie E. Nwankpa, Winifred l.Ijomah, Anthony Gachagan, and Stephen Marshall, "Activation Functions: Comparison of Trends in Practice and Research for Deep Learning", 2nd International Conference on Computational Sciences and Technologies, December 2020.