Automatic Recognition of Peruvian Car License Plates

Sebastian Escalante* and Victor Murray*†‡
*Department of Electrical Engineering, Universidad de Ingenieria y Tecnologia - UTEC, Lima, Peru.
†Department of Electrical and Computer Engineering, University of New Mexico, New Mexico, USA.
‡John A. Paulson School of Engineering and Applied Sciences, Harvard University, Massachusetts, USA.

***Abstract*—Some of the applications of automatic license plate recognition algorithms are security, tracking, and toll collection. However, due to the lack of regulations on Peruvian car license plates, developing a robust system to detect them represents a challenge. The problem is not only related to different rules based on vehicle types or rule year inconsistencies but also that many vehicles on the road have degraded license plates or thick protectors that hinder the character recognition process. In this manuscript, we present an optimized method based on the *k*-nearest neighbors (*k*-NN) classification method, which is compared with regular *k*-NN and multiclass support vector machines to recognize Peruvian car license plates, with accuracy results bigger than 96% for the optimized *k*-NN method.**

***Index Terms*—Automatic license plate recognition, *k*-nearest neighbors**

I. Introduction

Automatic license plate recognition (ALPR) algorithms are a widely studied due to their use in many applications, such as automation of parking lots, security devices for police patrols, traffic control, and many more [1]. These algorithms have been in development since the 1960s and have been improved since then. In [2], an extensive analysis of several ALPR methods, until 2013, was performed, showing that the general procedure to recognize the character in a license plate can be summarized as shown in Fig. 1.

The process starts when an image of the vehicle is introduced in the system. Then, different algorithms are used to detect the number plate; this step is done to extract the location of the plate, and, later, enhance it with a series of filters. Next, each character is segmented from the previous image, and optical character recognition methods are applied to identify each character [2].

Several studies have been carried out to develop ALPR algorithms adapted to the specific country codification where the research was conducted. However, there are few studies related to the detection of Peruvian vehicle license plates on the streets. In [3], in 2014, a system to recognize the plates with 95% accuracy on the entrance of a building was developed. The authors in [4] developed a method to recognize the plate's position with 100% accuracy regardless of the environment where the vehicle is located and a 23% error in character recognition considering the character arrangement for Peruvian license plates. Based on the previous, we can see that there is room for improvement in ALPR for Peruvian applications.

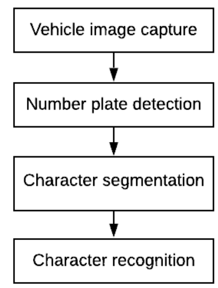


Fig. 1. General automatic number plate recognition process according to [2].

In this work, we present a first approach to improve character recognition on Peruvian license plates using two supervised machine learning (ML) methods: *k*-nearest neighbors (*k*-NN) and multiclass support vector machines (SVM). The methods described above use a license plate detection algorithm based on boundary characteristics that use a test set of 19 plates under different environmental conditions.

In the next sections, we describe the methods for license plate character extraction, training of the model using *k*-NN and multiclass SVM, and a small description of the license plate regulations in Peru in section II. Next, in section III, we present the results and a discussion about them. Finally, we present the conclusions and future work in section IV.

II. Methodology

A license plate detection algorithm based on boundary relation is used to detect the plates for later processing. This algorithm has its fundamentals on [5], which was based on Python-OpenCV coding. Besides, to perform the training of each model, five typographies are considered. The first one is directly obtained from original plates, and the characters are segmented manually using a vector graphic editor, and the other four are obtained from analyzing similar typographies to the first one. Then, three models are trained with these typographies, 180 characters in total. These models are based on *k*-NN and multiclass SVM, which are tested with the test set of plates. Later, a comparison is made with rules applied from Peruvian law regulations. Finally, the error is measured in the number of recognized characters in the test set of plates. In

978-1-7281-9377-9/20/$31.00 ©2020 IEEE

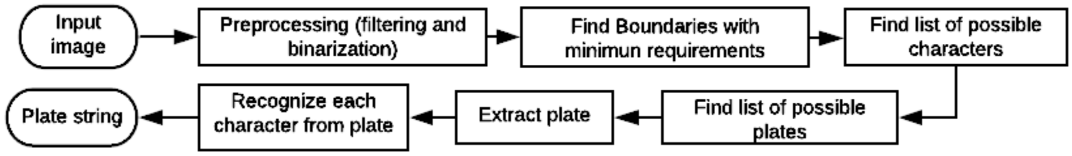
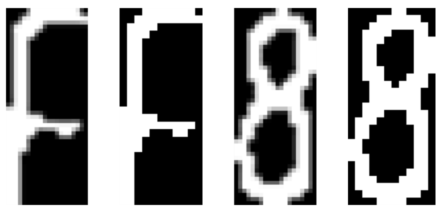


Fig. 2. Block diagram of the license plate character extraction.



(a) (b) (c) (d)

Fig. 3. Extracted characters from the license plate character extraction. (a) Resized ROI for character 'f'. (b) Segmented character for 'f'. (c) Resized ROI for character '8'. (d) Segmented character for '8'.

Fig. 4. Training characters used in each model. We have a total of 5 typefaces. Each typeface is composed of the 10 digits of the decimal system and the 26 letters of the alphabet.

the next subsections, we present a description of the extraction of license plate characters, how to train the model, and a summary of the Peruvian law regulations for license plates.

A. License plate character extraction

As a first step, it is necessary to recognize where the plate is and then extract each letter in order to apply different character recognition algorithms. The method described in Fig. 2 was chosen as our approach to license plate recognition algorithms. First, the input image is converted to a gray-scale and preprocessed by top-hat and bot-hat morphological operations to enhance details with a 3×3 square structural element. Then, an adaptive Gaussian threshold is applied to the result for binarization.

Afterward, a Canny edge filter is applied to find boundaries, and each of them is compared with a minimum value requirement of aspect ratio and area. Once the smallest boundaries are eliminated, the remaining are stored on a list of possible characters and compared with their neighbors to find relations of angle, changes in the area, width, height, and diagonal size. The ones that have formed groups of similar characteristics are stored on a list of possible plates. Finally, the largest group of possible characters on the set of possible plates is selected as the extracted plate and used as an input to a trained model of character recognition as Fig. 3 shows.

B. Training of the model

The training of all the models is based on five typefaces displayed in Fig. 4, this means that the training was done with 180 characters of different groups. Only the first typography was replicated as close as possible from the original typography used on vehicle license plates. The other five were chosen by hand using typography recognition online software in [6].

Several photos were taken in order to obtain the original shape of numbers in license plates. Each character was manually segmented as part of the process, as shown in Fig. 5.

Fig. 5. Manual segmentation of the character of a license plater, for training purposes. We have covered in red the first character for privacy reasons.

Note that we have covered in red the first character for privacy reasons. This process was repeated for each character until the completion of every alphanumeric character used in license plates.

From Fig. 4, two groups were extracted, the first one contains the string value of each alphanumeric symbol in plates, and the second one has the corresponding vectored image with a size of 840×1. This procedure is shown in Fig. 6, where the model block is a general representation for the several ML classifications applied to the recognized characters in the subsection II-A.

C. K-nearest neighbors algorithm

This model was trained as described in Fig. 6. k-NN is usually chosen as a benchmark because of its simplicity and usefulness. Two critical parameters to be considered when using this model are the distance measurement between values on the set and the number of neighbors that determines the output. Two types of k-NN were used: (i) a k-NN type with one neighbor and euclidean distance, and (ii) an optimized k-NN type with one neighbor and correlation distance. The correlation parameter was chosen based on the automatic fitting performed in MATLAB.

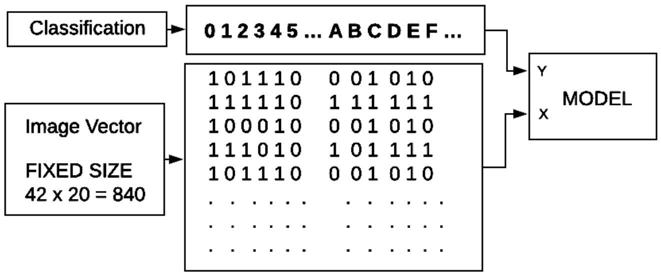


Fig. 6. Training procedure, from the vectors of characteristics to the final model.

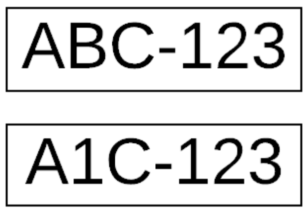


Fig. 7. License plate regulations in Peru for small and private vehicles but not for trucks.

D. Multiclass support vector machine

Support vector machines (SVM) are known for being an excellent option for binary classification. However, this cannot be used to solve multiclass classification problems by its own, in this case, multiple characters.

Error-Correcting Output Codes (ECOC) [7] is one way around this problem. This algorithm uses a coding matrix representing every character on the alphabet by using a classifier for each letter and returns a binary number. After the code is computed for a given letter, it is compared with all the others in the matrix. The system selects the code with the fewer error. In total, three models were trained.

E. License plate regulation in Peru

Character arrangement on the license plate is regulated by law. The details for the case of Peru can be found at [8]. These descriptions can be used in the ALPR to correct the most prominent cases of failure. In Fig. 7, it is shown that the last three digits are always numbers, and the first and third digits are always a letter. More rules apply for different types of vehicles, but in this case, the focus is on small vehicles and personal vehicles, other regulations are applied to trucks and buses.

When recognizing characters in a letter position, the most common causes of failure occur on zeros and letter O, which are very similar in shape with small differences on edges. Another case is the letter B which is usually misinterpreted as an eight. On the other side, when a number is taken by as a letter, this can happen with 1's changed by J's and B's by 8s.

III. Results

We took nineteen pictures of Peruvian license plates in different environments and angles, such as streets, parking lots, on the road, under sunlight, and without it. Thus, we have worked with 114 characters. In Fig. 8, we show an example of the step by step process to extract the characters as explained in subsection II-A. We can see in each image how the process, summarized in Fig. 2, is performed from detecting the location of the plate to extracting the characters.

The number of correctly recognized characters measures the performance of each model. In Table I, it can be observed the number of hits that every model made. The first column shows the alphanumeric characters, the second, the number of characters on the test set, and from third to the fifth column, the number of matches for each model. The best of the three models was the optimized *k*-NN, and the worst the *k*-NN, with 86.84% and 81.58%, respectively.

In Table II, it can be observed the misses for the models when trying to find the character in the first column. The models fail only characters 0, 2, 7, B, D, I, J, N, O, V and W, not in the remainder.

From an observation of the errors on each plate, the following rules were applied, with the restrictions and regulations explained in subsection II-E:

- If it is in a number position:
  - Replace O with 0 (zero).
  - Replace I with 2.
- If it is in a letter position:
  - Replace 1 with I.
  - Replace 5 with N.
  - Replace 0 with O.
  - Replace Y with W.

In Table III, the results of applying the rules improve the accuracy of each model near to 10%. These rules fixed the errors that cannot be done due to the regulation on plates. This fix maintained the optimized *k*-NN model as the best model of the three presented.

Also, the misses of each model are computed and presented in Table IV, where the most common mistake between models is the zero character with the number 8, followed by misses recognizing characters 7 and D.

The optimized *k*-NN model had the best overall performance recognizing characters from the test set, and its accuracy improved considerably when the rules were applied. Incorrect character recognition was mainly due to light conditions, blurring on the plate's image and orientation. Since no angle detection is applied to correct the plates' inclination, errors due to this condition influenced the results.

There are many open challenges to have a fully automatic license plate recognition system in Peru. In Fig. 9, we can see an example of a typical license plate is a challenge for our system. Note that we have covered in red the first character for privacy reasons. First, we can see how the black frame, added by the user, connects with the characters. These characters were original of black color, but due to no maintenance, the color is almost gone. Two problems appear: (i) where the character is still black, the frame is connected to the character, that implies the difficulty for a character segmentation, and (ii)

TABLE I

ACCURACY WITH BEFORE APPLYING THE RULES USING k-NN, OPTIMIZED k-NN AND A MULTICLASS SVM.

Char.	***k*-NN**	**Opt. *k*-NN**	**MCSVM**
Total: 114	93	99	94
Accuracy (%)	81.58%	86.84%	82.46%

TABLE II

MISSED CHARACTERS BEFORE APPLYING THE RULES USING k-NN, OPTIMIZED k-NN AND A MULTICLASS SVM.

Char.	***k*-NN**	**Opt. *k*-NN**	**MCSVM**
0	’OOOO 8 O’	’OO 8 O’	’8888’
2	’ 7’	’ I’	’ 7’
7	’ Y’	’ 1’	’ 1’
B	’888 6 8 8 8’	’888 8’	’888 6’
D	’ 8’	’ O’	’ B’
J	”	”	’ 3’
N	’ 5’	’ 5’	’M5’
V	”	”	’M’
W	’ Y’	’ V’	’ M’

at least 3 of the shown characters are now almost at the same color of the background, converting the character to an image with online a borderline. A multiscale texture analysis could help to extract this in other applications like [9].

IV. CONCLUSIONS AND FUTURE WORK

Three machine learning models were trained based on the original typography used on Peruvian license plates. The k-NN obtained the best recognition accuracy with one neighbor and correlation distance parameters. Then, after analyzing the output errors, rules were applied to the three models' outputs, and an improvement of almost 10% was obtained in the best model, getting results bigger than 96%. Due to the small test set, not all characters were equally evaluated. Moreover, the error analysis indicates that the angle inclination of characters should be taken into consideration and blurring on images and light conditions. Future work would involve providing a more extensive test set with all the alphanumeric characters included, detecting inclination, and developing a method to standardize image conditions before entering the system.

TABLE III

ACCURACY BY APPLYING THE RULES USING k-NN, OPTIMIZED k-NN AND A MULTICLASS SVM.

Char.	***k*-NN**	**Opt. *k*-NN**	**MCSVM**
Total: 114	108	110	103
Accuracy (%)	94.74%	96.49%	90.35%

TABLE IV

MISSED CHARACTERS BY APPLYING THE RULES USING k-NN, OPTIMIZED k-NN AND A MULTICLASS SVM.

Char.	***k*-NN**	**Opt. *k*-NN**	**MCSVM**
0	’ 8 O’	’ 8’	’8888’
2	’ 7’	”	’ 7’
7	’ Y’	’ 1’	’ 1’
B	’ V’	”	”
D	’ B’	’ O’	’ B’
J	”	”	’ 3’
N	”	”	’ M’
V	”	”	’M’
W	”	’ V’	’ M’

REFERENCES

[1] S.-L. Chang, L.-S. Chen, Y.-C. Chung, and S.-W. Chen, "Automatic license plate recognition," *IEEE transactions on intelligent transportation systems*, vol. 5, no. 1, pp. 42–53, 2004.

[2] C. Patel, D. Shah, and A. Patel, "Automatic number plate recognition system (ANPR): A survey," *International Journal of Computer Applications*, vol. 69, no. 9, 2013.

[3] G. A. J. Espinoza-Vásquez, "Sistema de reconocimiento de patrones en placas vehiculares para el acceso automático de visitas a un edificio," Pontificia Universidad Católica del Perú, Tech. Rep., Nov. 2014.

[4] G. A. Mundaca-Vidarte, "Detección de caracteres de placas de automóviles mediante técnicas de visión artificial," Universidad de Piura, Tech. Rep., Jun. 2016.

[5] Dahms, "Opencv3 license plate recognition python," https://github.com/MicrocontrollersAndMore/OpenCV_3_License_Plate_Recognition_Py 2016.

[6] MyFonts Inc., "What the font." [Online]. Available: https://www.myfonts.com/WhatTheFont/

[7] E. B. Kong and T. G. Dietterich, "Error-correcting output coding corrects bias and variance," in *Machine Learning Proceedings*, A. Prieditis and S. Russell, Eds. Morgan Kaufmann, 1995, pp. 313 – 321.

[8] Asociación Automotriz del Perú AAP, "Nueva placa única nacional de rodaje." [Online]. Available: https://aap.org.pe/placas/tipos/

[9] C. P. Loizou, V. Murray, M. S. Pattichis, M. Pantziaris, A. N. Nicolaïdes, and C. S. Pattichis, "Despeckle filtering for multiscale amplitude-modulation frequency-modulation (am-fm) texture analysis of ultrasound images of the intima-media complex," *International journal of biomedical imaging*, vol. 2014, 2014.

Fig. 8. Example of the step by step process to extract the characters as shown in Fig. 2. First, potential locations of the plate are found. Once the algorithm determines the right location, the characters are extracted.

Fig. 9. Typical degraded license plate in Peru with almost no maintenance. We have covered in red the first character for privacy reasons. Note how the frame connects with the characters. At least 3 of the shown characters lost their color and became only symbols with a borderline.