Improving the License Plate Character Segmentation Using Naïve Bayesian Network

Abdenebi Rouigueb[1][a], Fethi Demim[2][b], Hadjira Belaidi[4][c], Ali Zakaria Messaoui[3][d], Mohamed Akrem Benatia[1][e] and Badis Djamaa[1]

[1]*Artificial Intelligence and Virtual Reality Laboratory, Ecole Militaire Polytechnique, Bordj El Bahri, Algiers, Algeria*

[2]*Guidance and Navigation Laboratory, Ecole Militaire Polytechnique, Bordj El Bahri, Algiers, Algeria*

[3]*Complex Systems Control and Simulation Laboratory, Ecole Militaire Polytechnique, Bordj El Bahri, Algiers, Algeria*

[4]*Signals and Systems Laboratory, Institute of Electrical and Electronic Engineering, University M'hamed Bougara of Boumerdes, Boumerdes, Algeria*

rouigueb.abdenebi@gmail.com, demifethi@gmail.com, ha.belaidi@univ-boumerdes.dz, alizakariamessaoui@gmail.com,

Keywords: License Plate, Character Segmentation, Naïve Bayesian Network, DTW, CNN.

Abstract: Character segmentation plays a pivotal role in automatic license plate recognition (ALPR) systems. Assuming that plate localization has been accurately performed in a preceding stage, this paper mainly introduces a character segmentation algorithm based on combining standard segmentation techniques with prior knowledge about the plate's structure. We propose employing a set of relevant features on-demand to classify detected blocks into either character or noise and to refine the segmentation when necessary. We suggest using the naïve Bayesian network (NBN) classifier for efficient combination of selected features. Incrementally, one after one, high computational cost features are computed and involved only if the low-cost ones cannot decisively determine the class of a block. Experimental results on a sample of Algerian car license plates demonstrate the efficiency of the proposed algorithm. It is designed to be more generic and easily extendable to integrate other features into the process.

1 INTRODUCTION

ALPR systems (Du et al., 2013) have become crucial in various real-world applications, such as parking access control, road traffic monitoring, and tracking stolen vehicles. A typical ALPR system comprises four main modules: image acquisition, license plate localization (PL), character segmentation (CS), and character recognition (CR)(Du et al., 2013).

Compared to PL and CR, most recognition errors at the ALPR pipeline end occur in CS due to difficulties in correctly partitioning the plate image into blocks containing individual characters (He et al., 2008; Pan et al., 2008). The segmentation module takes the filtered-normalized plate image as input and then attempts to determine the smallest image segments expected to contain a single character each. For a comprehensive recent review of CS methods and their types in ALPR systems, refer to (Mufti and Shah, 2021; Du et al., 2013).

The three most popular existing CS approaches are horizontal-vertical projection (Jagannathan et al., 2013), connected component analysis (CCA) (Yoon et al., 2011), and character detection using deep neural networks which represent the state-of-the-art trend. For the last approach, the convolutional neural networks (CNN) architectures are the most used for CS. For instance, the models YOLO (Redmon et al., 2016), Faster-RCNN (Ren et al., 2015), and Classification-Regression Network (CR-NET) (Montazzolli and Jung, 2017) are used for character detection and recognition in real-world ALPRs systems. It is well to point out that all these methods are flawed and not universal. Furthermore, each method has its limitations, advantages, and appropriate conditions of use. See (Mufti and Shah, 2021) for exploring their respective performance summary, pros, and cons.

For some real conditions, these standard CS tech-

[a] https://orcid.org/0000-0001-5699-2721
[b] https://orcid.org/0000-0003-0687-0800
[c] https://orcid.org/0000-0003-2424-626X
[d] https://orcid.org/0000-0001-5753-5776
[e] https://orcid.org/0000-0003-1779-2705

Rouigueb, A., Demim, F., Belaidi, H., Messaoui, A., Benatia, M. and Djamaa, B.
Improving the License Plate Character Segmentation Using Naïve Bayesian Network.
DOI: 10.5220/0012091500003543
In *Proceedings of the 20th International Conference on Informatics in Control, Automation and Robotics (ICINCO 2023) - Volume 2*, pages 61-68
ISBN: 978-989-758-670-5; ISSN: 2184-2809

niques are not accurate enough when applied solely because they do not profit from the entire prior information about the license plate and the characters' structures. Indeed, for instance, characters in ALPR systems have almost the same size; lie on one or two straight horizontal layers; and pertain to a finite alphabet set, etc. Therefore, sophisticated ALPR systems in practice are those that can improve significantly CS by involving, if necessary for difficult cases, a part of the prior information. To achieve this goal, we need to precisely evaluate the confidence (i.e. probability) that a detected block corresponds to a useful character, then add the appropriate post-processing only for the confused blocks. Doing so helps to optimize the trade-off between the accuracy and the time of CS. In this context, this paper addresses the character segmentation problem by combining one of these standard techniques with license plate preliminary information using an NBN classifier.

The paper is structured as follows. The next section introduces the NBN classifier. Then, Section 3 describes the proposed algorithm and presents some suggested features, such as the dynamic time-warping (DTW) distance. In this section too, we show and discuss the obtained results. Finally, we comment on some conclusions, and we state future works.

2 NAÏVE BAYESIAN NETWORK CLASSIFIER

NBN classifier (Mittal et al., 2007) is a simple and powerful Bayesian network. Even with the restrictive independence assumption among the features, it yields competitive performance against other state-of-the-art classifiers. When NBN is used for classification, feature variables x_i are assumed to be *conditionally independent* given the class variable C. Its respective structure is shown in Fig. 1, in which the conditional joint probability $P(x_i, x_j|C)$ can be factorized into a product as follows:

$$P(x_i, x_j|C) = P(x_i|C)P(x_j|C), 1 \leq i \neq j \leq n. \quad (1)$$

NBN classifier predicts a new data point as the class with the highest posterior probability using the following formula:

$$\omega = argmax_{c_i} P(C = c_i) \prod_{j=1}^{n} P(x_j|C = c_i). \quad (2)$$

Bayesian networks, including NBNs, can deal immediately with missing features (not yet calculated). By using only the available features set $Y \subset X = \{x_1, ..., x_n\}$ (X all considered features), the classification rule is as follows:

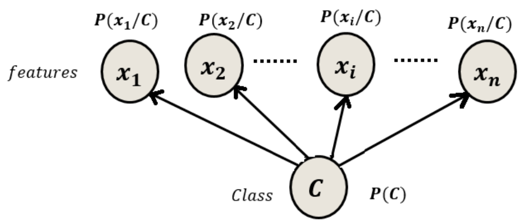


Figure 1: Naïve Bayesian network classifier structure.

$$\omega = argmax_{C_i} P(C = c_i) \prod_{x_j \in Y} P(x_j|C = c_i). \quad (3)$$

To construct the NBN, we need to discretize all features' variables. The distributions $P(C)$ and $P(x_i|C)$ $i = 1..n$ can be fixed by the user or estimated using the training data.

3 PROPOSED SOLUTION

3.1 Segmentation Algorithm

Our main contribution consists of performing a standard CS technique such as CCA, projection, or CNN. Then, NBN is incrementally utilized on request to refine the CS split positions using prior information by computing some specific features. The global scheme of the proposed algorithm is depicted in Fig. 2. Since our work focuses only on improving CS, we performed PL by manually double-clicking on the four edges of each plate. Hence, all eventual errors will be due to the CS stage, not plate extraction. In the pre-processing phase (step (a) of the algorithm shown in Fig. 2), a geometrical normalization is carried out by finding an *affine transformation* (Marcel, 1987) from the localized plate I_1 to a 30×140 pixels horizontal fixed-size non-tilted plate I_2. Indeed, we have $I_2 = A \times I_1 + B$ where the transformation parameters, the 2×2 matrix A and the 1×2 vector column B, can be estimated using only three among the four edge correspondences. Since the distance separating the camera from the car plate is so important compared to the car plate size, straight lines seem to remain straight, and parallel lines seem to remain parallel, but rectangles may change into parallelograms. It's worth remembering that affine transformation is suitable for such a kind of image shearing. After that, pixel intensity is normalized into the range $[0-255]$ grey color to reduce the effect of the variation in lighting conditions. Then, the Sauvola adaptive threshold (Sauvola and PietikaKinen, 2000) is used for binarization. Next, a morphological opening is applied to remove small objects from the image while preserving the shape and the size of large ones. Finally, simple horizontal and vertical projections can be used to eliminate the eventual remaining borders resulting from the lack of precision in the PL step. It is worth noting that binarization is not recommended (may be skipped) if deep neural networks are used for character detection in the next step.

In step (b) of the algorithm shown in Fig. 2, we propose to perform an initial segmentation using one of the state-of-the-art CS methods such as CCA, projection, or deep neural networks.

Objects (blocks) found at this level can correspond to either characters or non-characters (noise decorative patterns, etc). Prior information, such as the shape, size, position, etc., of characters in license car plates is necessary to confirm reliably the class (Character or non-character) of each block.

In this study, for illustrative purposes only, we have proposed to use the 07 following features of each detected block B: $x_1 = Height(B)$, $x_2 = width(B)$, $x_3 = MeanPixelIntensity(B)$, $x_4 = allign(B)$ (the number of blocks horizontally aligned with B), $x_5 = HeightConfirmedCh(B)$, $x_6 = min(2 - DDW(B, \lambda_i))$ with $\lambda_i \in$ {templates of the characters':'0',..., '9'}, and $x_7 = CNN(B)$ (the classification score of B into character|non-Character classes). x_5 is equal to the normalized difference between the height of B and the mean height of the blocks already confirmed as characters. x_7 is computed using a CNN model (the model VGG16 is used in our case).

Previously, before running this algorithm, the domain of features is discretized such that values within the same discrete value contribute roughly in the same manner in prediction. For example, for x_1, the three proposed values $1, 2$ and 3 correspond to the intervals $[18\ 25]$, $[14\ 17] + [26\ 29]$, and $[0\ 13] + [30\ +\infty]$, respectively. Most characters have $x_1 = 1$, and most non-characters have $x_1 = 3$. Then conditional probability tables $P(x_i/C)$ are estimated using training labeled data.

NBN estimates the more-likely class of each block given only its calculated features. In step (d) of the algorithm shown in Fig. 2, we start by determining the next interesting feature to be computed in terms of its cost and its relevance. Let F be the chosen feature and H the other features computed in already iterations. $X = \{F\} \cup H$ are used to infer the posterior probability $P^* = P(C = Char|X)$ using the following rule:

$$P^* = P(C = Char|X) = \frac{1}{Z} P(C = Char) \prod_{x_j \in X} P(x_j|C = Char), \quad (4)$$

with $P(C = Char)$ the prior probability that B is a character, and the evidence $Z = p(X) = P(C = Char)p(X|C = Char) + P(C = Non - Char)p(X|C = Non - Char)$ is a scaling factor.

Thus, P^* will be high for characters and low for non-characters. Then, P^* is compared with two decision pre-set thresholds T_1 and T_2 as follows: if $P^* > T_2$ then B corresponds to a character and if $P^* < T_1$ then B corresponds to a non-character, otherwise (if $T_1 < P^* < T_2$) the class of B can not be confirmed using only $\{F\} \cup H$.

The undecided class corresponds to the confusing difficult objects, which require additional verification. Here, two scenarios are conceivable. Either there is an error in the current segmentation. In this case, we must revise the segmentation using devoted methods (go from step (e) to step (c) in the algorithm shown in Fig. 2). Either the calculated attributes are insufficient to confirm the final class. In this second case, we suggest calculating and exploiting another feature that has a high cost but may be more relevant such as x_6 and x_7 (go from step (e) to step (d) of the algorithm in Fig. 2).

We should note that the low-cost features $\{x_1, x_2, x_3, x_4, x_5\}$ suffice to decide with confidence on most blocks; additional re-segmentation or high-cost features will still be demanded to deal with just a small number of difficult objects. As a demonstration, we have designed and implemented a classification algorithm that checks if the block B corresponds to connected characters. This verification algorithm is solicited in step (e) of the algorithm shown in Fig. 2.

3.2 Proposed Features

The proposed features $x_1, ..., x_5$ can be computed using simple formulas. Thus, in this section, we introduce the 02 high-cost features x_6 (VGG16 score), and x_7 (2D-DTW image matching distance).

3.3 VGG16 Score

VGG16 (Simonyan and Zisserman, 2014) is a simple-fast-accurate CNN model for image classification. Its architecture is depicted in Fig. 3. See the survey (Shashirangana et al., 2020) to cover the proposed deep learning methods, including the VGG variants for license plate recognition.

In this work, the blocks' images are resized to 224x224 pixels, so they match exactly with the VGG16 input shape. Blocks extracted from the training images are visually checked and labeled as "character" or "non-character" and then used to carry out a transfer learning of the base model VGG16.

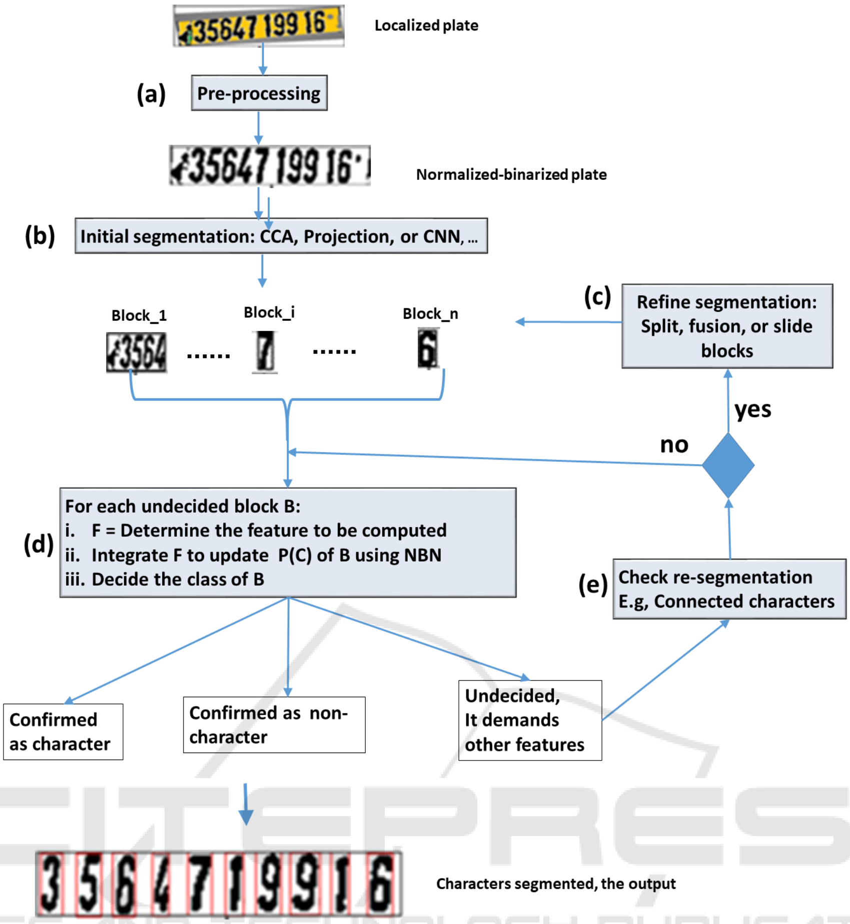


Figure 2: Character segmentation algorithm.

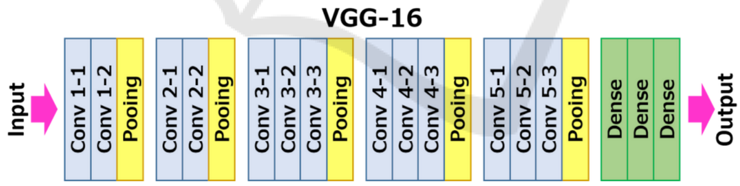


Figure 3: VGG16 architecture.

3.4 Dynamic Time Warping

3.4.1 One-Dimensional Dynamic Time Warping

DTW (Berndt and J.Clifford, 1994) is a distance usually used for sequence matching and optimal alignment. The distance $DTW(X,Y)$ between two sequences $X = (x_1, x_2, ..., x_n)$ and $Y = (y_1, y_2, ..., y_m)$ can be calculated by the construction of an $n-by-m$ matrix D of alignment, as shown in Fig. 4. The challenge consists in finding the optimal path from cell $(1,1)$ to cell (n,m) having the minimal average cumulative cost. Each cell (i,j) contributes $cost(x_i, y_j)$ to the cumulative cost. The cost function $cost(x_i, y_j)$ may be defined according to the application.

To perform computation efficiently, we can first initialize $D(1,1)$ to $cost(1,1)$. Then, the optimal path can be calculated using dynamic programming by employing the recursive equation shown in Eq. (5):

$$D(i,j) = Cost(i,j) + min \begin{cases} D(i-1,j) \\ D(i,j-1) \\ D(i-1,j-1) \end{cases} \quad (5)$$

The three previous stages, $D(i-1,j)$, $D(i,j-1)$, and $D(i-1,j-1)$, are assumed to be evaluated, and their results are stored when computing $D(i,j)$. Notice that the rest of the cells in the first column and

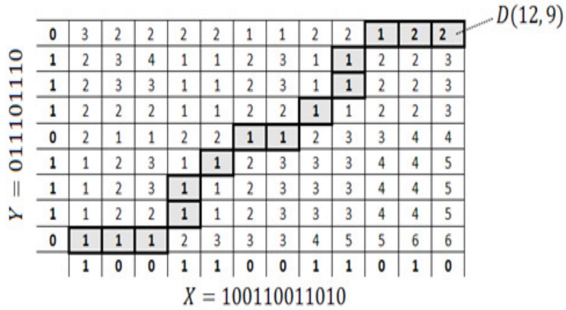


Figure 4: 1D-DTW example.

the first row have just one previous stage. In doing so, DTW tolerates one-many and many-one matching and allows thus to align sequences with different lengths. Moreover, DTW can be speeded by limiting the exploration to cells close to the diagonal $(1,1) \longleftrightarrow (n,m)$.

In the example depicted in Fig. 4, the function $cost(x_i, y_j) = abs(x_i, -y_j)$ is considered. The optimal alignment path, between sequences X and Y, is highlighted on the matching matrix, where its length is 14, and its cumulative cost is 2.

3.4.2 Two-Dimensional Dynamic Time Warping

Indeed, extending 1D-DTW to 2D-DTW is so interesting to match elastically images, which means that the alignment should be invariant to image resizing and geometrical transformations that preserve local neighborhood features. In this section, we present a concise introduction to 2D-DDW (Lei and Govindaraju, 2004), a variant of 2D-DTW for image matching.

In a similar way, we can define $D(i_1, j_1, i_2, j_2)$ as the average cumulative cost matching between the partial image $(1:i_1, 1:j_1)$ of image I_1 and the partial image $(1:i_2, 1:j_2)$ of image I_2 as shown in Eq. (6):

$$D(i_1, j_1, i_2, j_2) = \begin{array}{l} min_{k:1..15}\{Cost_k(i_1, j_1, i_2, j_2) + \\ PreviousStage_k\} \end{array} \tag{6}$$

Table 1 displays the possible values of the previous stages and their respective costs. 15 previous stages are involved in the two-dimensional model of Eq. (6) instead of 03 previous stages in the one-dimensional model illustrated in Eq. (5). The cost $Cost_k(i_1, j_1, i_2, j_2)$ is not the same for all stages as in the 1D-DTW case, but it is defined depending on the selected previous stage k. We have used the function cost (Lei and Govindaraju, 2004) especially proposed for image matching.

Table 1: Computing of $Cost_k(i_1, j_1, i_2, j_2)$ in the 2-DDW recurrence rule.

k	**PreviousStage**$_k$	**Cost**$_k(i_1, j_1, i_2, j_2)$
1	$D(i_1-1, j_1-1, i_2-1, j_2-1)$	$R+C$
2	$D(i_1-1, j_1-1, i_2-1, j_2)$	$R+C$
3	$D(i_1-1, j_1-1, i_2, j_2-1)$	$R+C$
4	$D(i_1-1, j_1-1, i_2, j_2)$	$R+C$
5	$D(i_1-1, j_1, i_2-1, j_2-1)$	$R+C$
6	$D(i_1-1, j_1, i_2-1, j_2)$	R
7	$D(i_1-1, j_1, i_2, j_2-1)$	$R+C$
8	$D(i_1-1, j_1, i_2, j_2)$	R
9	$D(i_1, j1-1, i_2-1, j_2-1)$	$R+C$
10	$D(i_1, j1-1, i_2-1, j_2)$	C
11	$D(i_1, j1-1, i_2, j_2-1)$	C
12	$D(i_1, j_1-1, i_2, j_2)$	$R+C$
13	$D(i_1, j_1, i_2-1, j_2-1)$	$R+C$
14	$D(i_1, j_1, i_2-1, j_2)$	R
15	$D(i_1, j_1, i_2, j_2-1)$	C

where $R = 1D\text{-}DTW(I_1(i_1,:), I_2(i_2,:))$ and $C = 1D\text{-}DTW(I_1(:,j_1), I_2(:,j_2))$

Table 2: Segmentation results of the test images by the proposed algorithm.

Real class	Predicted class Character	Non-character	Undecided
Character	2950	5	26
Non-character	4	10	5

Table 3: Segmentation results of the test images by the YOLOv5 model.

Real class	Predicted class Character	Non-character	Undecided
Character	2963	9	9
Non-character	7	7	5

3.5 Experimental Results

A series of experiments are carried out on a dataset including 850 vehicle images with Algerian license plates, each measuring 600x840 pixels. 300 images drawn randomly are used for the test; the rest (550) are used to estimate the NBN and the VGG16 model parameters.

The CS test results using our method are checked visually, a summary is given in Table 2. The rate of correct segmentation is $(2950+10)/(2950+5+26+4+10+5) = 98.67\%$ which is accurate enough compared with the quality of images.

We repeated the same tests on the same images using the YOLO-v5 model (Jocher, 2020). YOLOv5 is a fast and accurate CNN model for detecting objects in images, it is known as one of the leading solutions for this task. For its use, we retrained it using the training sample containing 550 labeled images. The test results obtained are displayed in Table 3. For the prediction, following the same approach as the proposed solution, we retrieved from YOLO output the probability $P(B = Char)$ that a detected block B is character. Then, the block B is assigned to the "undecided class" if $P(B = Char)$ is between two thresholds T_1, T_2 (i.e. non-confirmation case).

In terms of accuracy, the results in Tables 2 and 3 are quite similar, with a slight advantage to YOLOv5. Our method tends to struggle with separating connected characters, often assigning them to the undecided class. Conversely, YOLOv5 sometimes fails to recognize new non-character objects and incorrectly classifying them as characters or undecided.
Both algorithms are tested on the same workstation (CPU: Intel i5 3.7GHz, RAM:32Go). In terms of speed, the average time to process an image of a plate is 32ms for the proposed method and 45ms for YOLOv5. The proposed method is faster than YOLOv5 on the CPU.

Table. 4 illustrates the segmentation and recognition (OCR) outputs of a set of localized plate examples using the proposed method. A deep visual analysis of these results has revealed the following facts:

1. the majority of errors of kind *non-character predicted as Character* corresponds to decorative or publicity drawings often laid in particular places.
2. the majority of errors of kind *Character predicted as non-character* corresponds to characters which are excessively damaged or which are transcribed in a font too different from that one of the used characters prototypes.
3. The undecided objects usually correspond to linked characters, damaged characters, partial characters, or errors in the binarization step in the pre-processing stage.

Linked characters and partial characters problems often have their origin at the level of the preprocessing or the initial segmentation (steps (a) and (b) in Fig. 2). Our solution has the advantage of trying to classify the blocks into three classes instead of two. The third class corresponds to confused objects (undecided) that need more examination. For this case, we need to perform several checks to identify the type of an undecided block (step (e) in Fig. 2). Then, depending on that, our method can go back and revise the binarization or segmentation (step (c) in Fig. 2). In this step, we can split a block, merge two blocks, or resize a block. Otherwise, it can involve supplementary features (step (d) in Fig. 2).

To illustrate, we suggest adding a two-linked characters check module for the undecided objects. A block B has a high potential to embody two characters or more when $P(C = Char|x_1, x_3, x_4, x_6)$ is high, $P(C = Char|x_6, x_7)$ is low, and x_2 is high (i.e. large blocks). In this case, the block is split into two adjacent blocks B_{left} and B_{right} so that B_{left} has a high probability to be a unique character. The algorithm shown in Table 5, analyses the content of B and might split it using the robust 2D-DTW matching distance. It divides B at the horizontal position i if the distance 2D-DTW(B(2:i,:), character templates) is very low. Obviously, other checks can be easily integrated.

In our experiment, the two-linked check module has allowed us to split 18/29 connected characters successfully.

3.6 Complexity Analysis

The computation time of the 2-DWW score is significantly higher compared to the rest of the steps in our algorithm. In this section, we estimate the complexity of the proposed 2-DDW implementation.

Suppose both images I_1 and I_2 are N-by-N. The complexity of constructing the matching matrix by 04 nested loops is $\alpha = O(N^4)$ iterations. We need to determine the cost value of the 15 previous stages at each iteration. It is worth noting that a stage may be the previous of many stages; thus, speeding the computation is possible by saving and reusing intermediate cost values.

According to (Lei and Govindaraju, 2004), the evaluation of $Cost_k(i_1, j_1, i_2, j_2)$ involves the evaluation of 1D-DTW between rows $I_1(i_1, :)$ and $I_2(i_2, :)$ or between columns $I_1(:, j_1)$ and $I_2(:, j_2)$. In the worst case, 1D-DTW is computed once at most between each row pair and between each column pair of images I_1 and I_2. The complexity of computing all stages' costs is $\beta = 2O(N^2) \times O(1D\text{-}DTW) = O(N^4)$. The entire complexity is then $\alpha + \beta = O(N^4)$. One of our main contributions consists in decreasing the time complexity from $O(N^6)$ (Lei and Govindaraju, 2004) to $O(N^4)$ using the dynamic programming principle.

3.7 Future Improvement

In this work, we have illustrated a robust segmentation framework that quantifies the segmentation quality by assessing the probability that a block corresponds to a character and reducing the processing time. These two aspects are interesting to develop new ALPR systems analyzing video sequences where cars move continuously and the image acquisition conditions vary. This characteristic may help combine partial results of images covering identical vehicles efficiently and create real-time solutions.

We suggested predicting the class of each block among the three following labels: "confirmed as a

Table 4: Examples of character segmentation and recognition.

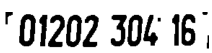
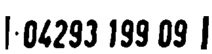
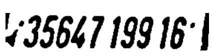
Localized plate	Pre-processing	Segmentation	Recognition	
			**0120230416**	**success**
			**0429319909**	**success**
			**0930010716**	**success**
			**3564779916**	**success**
			416431078	failed
			**1134610516**	**success**
			8516131861	failed
			**0994130016**	**success**
			**3346810816**	**success**

Table 5: Algorithm that checks and splits two-linked characters.

Input: B, the block to be checked and split.
Output: B_{left},B_{right} // B may be split into B_{left} and B_{right}

$K \leftarrow$ width of B;
for $i = 1$ *to K do*
  $D(i) \leftarrow min_{j:0..9}$ 2-*DDW*$(B(1:i,:),\lambda_j)$;
//λ_j: template of digit $'j'$
  $t \leftarrow$ index of $(min(D))$;
If $D(t) < threshold$, then % split B
  $B_{left} \leftarrow B(1:t,:)$;
  $B_{right} \leftarrow B(t+1:K,:)$;

character", "confirmed as a non-character", and "undecided". CS errors may be due to pre-processing, NBN classification, or undecided object processing steps deficiency. In future works, we recommend the following directions:

1. defining new cost-effective features to minimize the number of undecided objects. For example, global features on the order and distances between successive characters appear to be beneficial.
2. finding the best order of features for each block. For example, we can create a cascade of the low-cost classifiers as shown in the CASACRO algorithm (Hanczar and Bar-Hen, 2021).
3. examining various types of undecided blocks and suggesting suitable processing, including the possibility of re-segmentation.

4 CONCLUSION

In this study, we have developed a robust character segmentation method for vehicle license plates. Initially, we employ a standard CS technique such as CCA, projection, or CNN to identify the potential characters. Subsequently, we utilize the NBN classifier to predict the label of each block, categorizing them as "Character", "Non-Character", or "Undecided". In addition, the proposed algorithm provides the probability that a block corresponds to a character, which is valuable in various applications. However, different features can have varying impacts on the classification process, and their computation incurs different time costs. Therefore, the most computationally expensive features should be involved only for a few for challenging or ambiguous blocks.

We emphasize that, in the problem we're addressing, only a few low-cost features are computed for most blocks, and almost all the features are approximately independent conditionally on the class. Our primary objective is to strike a balance between accuracy and speed.

To meet these criteria, we recommend using the NBN classifier as it aligns perfectly with these assumptions and requirements. In this study, we have also introduced a faster variant of the 2-*DDW* template matching algorithm (Lei and Govindaraju, 2004), significantly reducing complexity to $O(N^4)$.

State-of-the-art CNN models, like YOLOv5, offer several advantages, including ease of implementation, high recognition rates, and running faster on GPU. They are challenging to surpass in these aspects. Nevertheless, the proposed method presents distinct advantages. It operates as a white box, where its modules and steps can be explained. Additionally, it runs efficiently on the CPU, and it can be enhanced in the future in various aspects.

We have introduced a general approach to CS. Our approach allows for the utilization and integration of segmentation methods published in the literature. The core principles of our proposal are as follows: i) We assess the probability that a block is either a character, not a character, or uncertain base on the pre-computed features. To accomplish this, we have chosen to employ Bayesian networks due to their capability to handle missing features and assess class probabilities in a formal manner. ii) The introduction of the "undecided block" class is significant, as it simplifies the process of either requesting additional features or adjusting the segmentation. iii) We can incorporate prior information regarding the structure of license plates. Thus, it is less sensitive to the over-learning problem often encountered for deep neural network models.

Overall, obtained partial results are promising. For instance, further improvements by providing new features and comparisons with state-of-the-art methods using international public datasets are recommended to highlight the proposed approach.

REFERENCES

Berndt, D. and J.Clifford (1994). Using dynamic time warping to find patterns in time series. In *Working Notes of the Knowledge Discovery in Databases Workshop*, pages 359–370.

Du, S., Ibrahim, M., Shehata, M., and Badawy, W. (2013). Automatic license plate recognition (alpr): A state-of-the-art review. *IEEE Trans. on Circuits and Systems for Video Technology*, 23:311–325.

Hanczar, B. and Bar-Hen, A. (2021). Cascaro: Cascade of classifiers for minimizing the cost of prediction. *Pattern Recognition Letters*, 149:37–43.

He, X., Zheng, L., Wu, Q., W.Jea, Samali, B., and Palaniswami, M. (2008). Segmentation of characters on car license plates. In *Proc.IEEE 10th Workshop on Multimedia Signal Processing*, pages 399–402.

Jagannathan, J., Sherajdheen, A., vijay deepak, R. M., and Krishnann, N. (2013). License plate character segmentation using horizontal and vertical projection with dynamic thresholding. In *IEEE Inter. Conf. on Emerging Trends in Computing, Communication and Nanotechnology*, pages 700–705.

Jocher, G. (2020). Yolov5 by ultralytics.

Lei, H. and Govindaraju, V. (2004). Direct image matching by dynamic warping. In *Computer Vision and Pattern Recognition Workshop*.

Marcel, B. (1987). *Geometry I*. Berlin: Springer.

Mittal, A., Kassim, A., and Tan, T. (2007). *Bayesian Network Technologies: Applications and Graphical Models*, chapter chapter 1, pages 1–12. IGI Publishing, Hershey New York.

Montazzolli, S. and Jung, C. (2017). Real-time brazilian license plate detection and recognition using deep convolutional neural networks. In *2017 30th SIBGRAPI conference on graphics, patterns and images (SIBGRAPI)*, pages 55–62. IEEE.

Mufti, N. and Shah, S. A. A. (2021). Automatic number plate recognition: A detailed survey of relevant algorithms. *Sensors*, 21(9):3028.

Pan, M.-S., Yan, J.-B., and Xiao, Z.-H. (2008). Vehicle license plate character segmentation. *Inter.Journal of Automation and Computing*, 05(04):425–432.

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

Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28.

Sauvola, J. and PietikaKinen, M. (2000). Adaptive document image binarization. *Patt. Recog.*, 33:225–236.

Shashirangana, J., Padmasiri, H., Meedeniya, D., and Perera, C. (2020). Automated license plate recognition: a survey on methods and techniques. *IEEE Access*, 9:11203–11225.

Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Yoon, Y., Ban, K.-D., Yoon, H., and Kim, J. (2011). Blob extraction based character segmentation method for automatic license plate recognition system. In *Proc.IEEE International Conference on Systems, Man, and Cybernetics*, pages 2192–2196, Korea.