License Plate Character Recognition Using Multiclass SVM

Amir Ebrahimi Ghahnavieh, Abolghasem A. Raie

Mobile Robots Research Laboratory, Faculty of Electrical Engineering, Amirkabir University of Technology, Tehran, Iran. amir.ebrahimi66@gmail.com

Abstract: This paper proposes a new method for license plate character recognition based on Support Vector Machines (SVMs). The goal is to design a fast, accurate and simple classifier for real-time applications. In proposed method first, most probable outputs are recognized by probabilistic models and then, final output is achieved by support vector machines. This design has simple training and obtains adequate speed and accuracy compared with other SVM-based systems. The accuracy is 95.54% and the needed time to process a whole plate is about 60 milliseconds.

[Ebrahimi A, Raie A. **License Plate Character Recognition Using Multiclass SVM.** *J Am Sci* 2012;8(1s):38-42]. (ISSN: 1545-1003). http://www.jofamericanscience.org. 6

Keywords: Character recognition; license plate recognition; SVM; probabilistic neural network; maximum likelihood

1. Introduction

This paper concerns about license plate character recognition. License plate recognition has lots of applications in intelligent transportation systems such as traffic control, fee payment, issuing fines, etc. Therefore, finding a way to increase the performance of such systems is needed. A license plate recognition system consists of three sub-systems: license plate detection, character segmentation and character recognition. Character recognition sub-system is the most important part which has direct effect on overall system and will be discussed in this paper.

Various methods have been used for character recognition in license plates so far. Template matching is a simple way for this purpose. Also, probabilistic neural networks showed better results compared with template matching (Y. Hu et al., 2005). Moreover, nowadays MLP neural networks are so popular (C. Anagnostopoulos et al., 2006 and A. Rahman et al., 2003), but in these networks lots of training parameters are involved and training process will be finished after obtaining the first satisfying separator hyper plane.

Support Vectors Machines, due to optimized separator, give higher generalization capability compared with neural networks, and are introduced as a superior classifier. Indeed, these kinds of classifiers are binary and have to be generalized for license plate character recognition purpose, which needs multi-class classifier. In (K. K. Kim et al., 2000) SVM is used for license plate character recognition, and a traditional method called "one-against-all" is used for generalization. This method does not work on various datasets necessarily. Also, there is another traditional method called "one-against-one" which is used for handwritten characters in (Renata F. P. Neves et al., 2011) and obtains 96% accuracy. Most important defect of this method is in computational complexity, which is proportional to square of number of classes. Another method is such that first, two most probable characters are selected by a neural network and then, final result is given by an SVM (A. Bellili et al. 2003).

Our aim in this article is to use advantages of SVMs and avoid problems of traditional methods to design a suitable classifier. The contribution of this paper is using a probabilistic approach to obtain this goal, which has a proper speed such as *one-against-all* and a proper accuracy such as *one-against-one.*

The remainder of this study is organized as follows: First, a review on SVM and some generalization methods for multi-class applications is stated. Then, proposed method is introduced and finally, experimental results are presented.

2. A Review on SVM and Multi-Class SVM

Support vector machines for the first time were introduced in (C. Cortes and V. Vapnik, 1995). So far, SVM is known as a proper and optimum classifier, which tries to separate classes using the most appropriate margin. Achieving the best margin, improves generalization ability of SVM, compared with other classifiers. SVM, in contrast to most classifiers, just uses a small part of data which is close to the separating margin. So, it is not sensitive to large training databases.

a) Linear SVM

In SVM, for binary classification in a *d*-dimensional feature space, a linear decision function is used:

$$f(x) = \omega \times x + b; \tag{1}$$

where ω is the weight vector and b is the bias. Classification is given by *y(x)=sgn(f(x))*. *y(x)* only has two outputs corresponding to each class; +1 and -1. In linearly separable case, discrimination function is a hyper plane which discriminates points of each class

(Figure 1). To obtain a canonical form of the hyper plane, ω and b are rescaled such that the points of two classes which are closest to the hyper plane satisfy $|\omega.x+b=1|$. Therefore, for all points of two classes, $\{(x_i, y_i), i=1,2,...,l\}$ with labels $y_i \in \{-1,+1\}$ we have two constraints:

$$\begin{cases}(\omega.x_i)+b \geq +1 \text{ if } y_i = +1 \\ (\omega.x_i)+b \leq -1 \text{ if } y_i = -1\end{cases} \tag{2}$$

ω is normal to the hyper plane which separates two classes. $b/\|\omega\|$ is the perpendicular distance from the hyper plane to the origin. And $\|\omega\|$ is the Euclidian norm of ω.

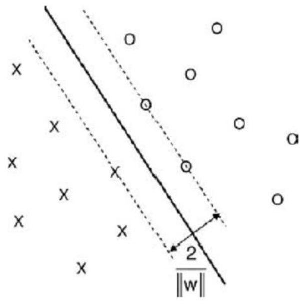

Figure 1. Class separation in SVM.

The margin value, in this case, is $2/\|\omega\|$. Maximizing the margin is the same as minimizing $(\omega.\omega^T)/2$ subject to the constraints. Minimizing this function can be achieved by using Lagrange multiplier:

$$f(x)=\sum_{i=1}^{l} y_i\ \alpha_i\ (x.x_i)+b \tag{3}$$
$$=(\sum_{i=1}^{l} y_i\ \alpha_i\ x_i)\ .\ x+b=\omega\ .\ x+b;$$

where α_i are Lagrange multipliers and $\omega=\sum_{i=1}^{l} y_i\ \alpha_i\ x_i$.

SVM is a binary classifier, and this feature is the greatest defect of it. In other words, this classifier can only set apart two classes and is unable to give a solution to multi-class problems. Because of this, some methods are suggested so far. Generally, there are two approaches for this problem. The first approach is to optimize all equations subject to all constraints and design a multi-class SVM. On the contrary, the second approach is to use some binary SVMs instead of a multi-class one. The former is too complex and time consuming, so it is not recommended. The following section discusses based on the latter approach.

b) Multi-Class SVM

As stated earlier, SVM is a binary classifier. Techniques to generalize it, based on the type of combination of binary classifiers, have different speeds and accuracies. Some common methods are mentioned below.

One-against-all

This is the simplest way to combine binary classifiers, and there is one classifier for each class which separates the class from others. In this method, samples of one class are considered as *class+1* and samples of the rest of the classes are considered as *class-1* for corresponding SVM. For *K* classes, *K* SVM classifiers are designed and all of them will be run for a test sample. Figure 2 shows this kind of separation.

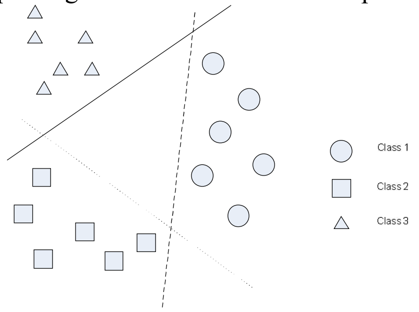

Figure 2. One-against-all method

One of the defects of this system is unbalanced training data. So, the number of training samples in *class+1* is very lower than *class-1*. After training each SVM for a test sample, outputs of all SVMs are studied. The outputs of SVMs can be confusing, because instead of one class, several classes might give positive result, or all of the classes might give negative result and a blind area is created (Figure 3).

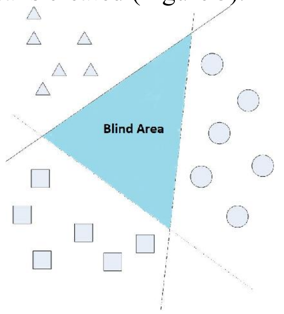

Figure 3. Blind area in one-against-all method

Because of this problem, in (Javad Sadri et al., 2003) the greatest output was chosen as the winner. Presenting the greatest output seems to be unfair and does not give the appropriate result necessarily. Therefore, in (Tiago C. Mota and Antonio Carlos G. Thome., 2009) some strategies according to histogram of SVM's outputs are introduced and MLP strategy shows the best accuracy among all.

One-against-one

Another technique is to use one SVM for each pair of classes which is used in (Renata F. P. Neves et al., 2011) for handwritten character recognition. For K classes, $K(K-1)/2$ SVMs are designed. Each input image will be evaluated by all SVMs and it is expected that the desired class wins more than other classes. Figure 4 shows the diagram of this method.

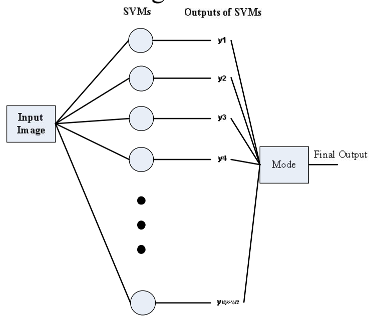

Figure 4. One-against-all method in (Renata F. P. Neves et al., 2011)

From authors' point of view, this method is more accurate than (A. Bellili et al., 2003). But, this solution is time consuming and costly. In addition, forcing all SVMs to give wrong answers might affect the final result. Also, it is vague when two classes have equal wins during competitions. Furthermore, the number of classifiers grows up sharply, according to the number of classifiers. This method might be inefficient for larger problems (John C. Platt et al., 2000).

3. Proposed Method

In (A. Bellili et al., 2003) for a test sample, two maximum outputs are selected as the most probable ones. And a test sample will be evaluated by an SVM classifier, if it exists. The major problem of this method which is not true, is considering outputs of a neural network as probabilities.

In suggested method, which is not used so far, first with a probabilistic classifier, most probable outputs are selected and then, a fore-trained SVM makes final decision for these outputs. This kind of training is simpler, conceptual and intuitive compared to MLP networks. Also the training parameters are less than MLP. In this paper, two kinds of probabilistic classifiers are stated; Probabilistic Neural Network (PNN) and Maximum Likelihood (ML) model.

a) *Probabilistic Neural Network*

In (Duda R. O. et al., 2000) an algorithm for PNNs is presented. Suppose that we have N patterns and each one has a d-dimensional feature vector according to K classes. PNN consists of d input units, and all of them are connected to all N pattern units. Each pattern unit is connected to one of K output units. First, each training sample x_{ji} is divided on ($\sum_{i=1}^{d} x_{ji}^2$) for normalization purpose. Then, network weights are equated to training samples:

$$w_{jk} = x_{jk}. \tag{4}$$

For a test sample, after normalization, the pattern units calculate inner product:

$$z_k = w_k^T.x \tag{5}$$

And then, emit a nonlinear function of z_k.

Here, function $e^{\frac{z_k-1}{\sigma^2}}$ with optimized value $\sigma = 0.26$ (using trial and error) is used. Each output unit sums the contributions from all pattern units connected to it. The class with larger output is selected as the winner. Test algorithm is shown in Table 1.

TABLE 1. PNN classification algorithm

```
PNN Classification
1 begin initialize $k = 0, x = test\ pattern$
2     do $k \leftarrow k+1$
3         $z_k \leftarrow w_k^T.x$
4         $g_c \leftarrow g_c + \exp(\,(z_k-1)/\sigma^2\,)$
5     until $k = K$
6     $class = \arg\max g_i(x)$
7 end
```

b) *Maximum Likelihood Solution*

Considering a K class problem, prior class probabilities are defined by $P(C_k) = \pi_k$ and general class-conditional densities $P(\varphi \mid C_k)$ where φ is the input feature vector. Supposing a training dataset $\{\varphi_n, t_n\}$ where $n = 1, \ldots, N$ and t_n is a binary target vector of length K that uses the *1-of-K* coding scheme, so it has components $t_{nj} = I_{jk}$ if pattern n is from class C_k. The likelihood function is given by:

$$P(\{\varphi_n, t_n\} \mid \{\pi_k\}) = \prod_{n=1}^{N} \prod_{k=1}^{K} \{P(\varphi \mid C_k)\pi_k\}^{t_{nk}}. \tag{6}$$

Taking the logarithm, setting the derivative with respect to π_k equal to zero with preserving the constraint $\sum_k \pi_k = 1$ and finally using Lagrange multiplier, we obtain (Bishop C. M., 2006):

$$\pi_k = \frac{N_k}{N}; \tag{7}$$

where N_k is the number of training samples related to class k. Now suppose that the class-conditional densities are given by Gaussian distributions with a shared covariance matrix, so that:

$$P(\varphi \mid C_k) = \mathbb{N}(\varphi \mid \mu_k, \Sigma); \tag{8}$$

where μ_k is the mean and Σ is covariance matrix of Gaussian distributions. Using Maximum Likelihood solution gives:

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^{N} t_{nk}\ \varphi_n; \tag{9}$$

and

$$\Sigma = \sum_{k=1}^{K} \frac{N_k}{N} S_k; \tag{10}$$

where

$$S_k = \frac{1}{N_k} \sum_{n=1}^{N} t_{nk} (\varphi_n - \mu_k)(\varphi_n - \mu_k)^T. \tag{11}$$

By acquiring μ_k and Σ , parameters of probabilistic model are completed.

So far, two probabilistic models have been introduced. Considering two larger outputs of these probabilistic models and assigning them to a fore-trained SVM related to the outputs, gives a proper accuracy. Experimental results will be presented in the following section.

3. Experimental Results

For 24 available classes consisting of 9 numerals and 15 alphabets, which is used in Persian license plates, we use 1200 samples (50 samples per each class) for training. Dataset is courtesy of Baninick Company, and is available with a request to http://www.baninick.com. All images are binary with the same size 20×12 . Zoning method is used to extract 15 features. Dataset consists of rotated, noisy and distorted characters. The process is done with a Core 2 Quad CPU 2.67 GHz and 3.5 GB RAM and MATLAB software.

For MLP-SVM method, a 15×20×24 structure is used. In Table 2, all classifiers are compared. For all classifiers, the same SVM parameters and the same training sets are used. One-against-all method gives adequate speed and One-against-one gives adequate accuracy. The difference in the rest of classifiers is in separating the most probable characters, before using SVM. As shown in the first column of Table 2, the training time of MLP neural networks is very large, compared with probabilistic models and shows that MLP-NN is just suitable for offline-training applications.

For test, 18945 characters are used. Testing time of each system is depicted in second column. PNN-SVM combination gives the fastest process speed, and along the one-against-all, has considerably difference with other methods. Also in third column, the accuracy of each system is illustrated, where ML-SVM combination partly gives the better accuracy compared to other methods. While the traditional one-against-all method with maximum output, gives unacceptable results on our dataset.

TABLE 2. Results

	Training time (s)	Testing time (s)	Accuracy (%)	Recognition time for each license plate (ms)
One-against-all	9.75	154.94	28.17	65.4
MLP-SVM	143.34	218.99	94.76	92.5
PNN-SVM	5.70	131.80	95.11	55.7
ML-SVM	6.18	229.17	95.54	96.8
One-against-one	5.61	5651.99	94.86	2386.7

Since each Persian license plate has 8 characters, it takes 8 times for a system to recognize the whole of a license plate, instead of just a single character. The average time to recognize a license plate is shown in the last column of Table 2. Hence, all of the methods can be used in a real-time application except one-against-one.

In our experiments in 97.75% of all cases, ML-SVM found the correct class in the first step. For PNN-SVM and MLP-SVM we obtained 96.74% and 96.48% respectively. So, the ML-SVM method was more successful to find the most probable classes.

In Table 3 misclassifications in all 3 methods are shown. Each row shows the correct class and each column shows the wrong selected class. As we can see, most of the mistakes was because of some especial characters such as (2,3), (3,4), etc. Indeed, characters {و،هـ،2،3،4،5،6،7} cause 86.7% of mistakes. A simple way to block these misclassifications is a post processing step for most confusing characters. This step can contain a new feature space instead of the prior one which considers finer details in the character. Finer details are the key to approach more accurate results.

Also, another application of Table 3 is in hierarchical structures. This table can be considered as a confusing matrix, which can be used to select proper macro-classes in a hierarchical structure. Indexes of the table can be selected as a similarity criterion between classes and macro-classes.

4. Conclusion

PNN networks can be trained in a fraction of the time it takes to train standard feed-forward networks. These networks have parallel structure, so parallel processing techniques make them faster to run. Moreover, Maximum Likelihood model gives good result, although its structure is simple. In this paper, we achieved proper accuracy and speed, by replacing MLP neural network with probabilistic models for license plates. In addition, selecting several probable outputs instead of two choices, gives better results by reducing the speed.

TABLE 3. Misclassification matrix

	1	2	3	4	5	6	7	8	9	الف	ب	س	د	ع	ق	هـ	ح	ل	م	ن	ص	ط	و	ی
1	0	4	0	0	1	0	9	0	9	0	0	1	0	0	0	9	0	0	0	0	0	0	8	0
2	0	0	446	209	0	0	95	0	1	0	1	6	0	1	3	0	26	0	5	2	0	0	9	0
3	0	109	0	499	0	0	5	0	9	0	0	1	1	0	0	1	4	0	9	0	0	2	0	3
4	0	2	29	0	0	0	0	0	0	0	0	0	0	5	0	2	0	0	11	0	0	0	2	0
5	0	1	1	0	0	0	0	0	0	0	0	1	54	24	46	146	0	0	2	2	0	0	5	0
6	0	9	0	60	0	0	13	29	1	0	0	4	0	53	0	0	0	0	11	0	3	0	442	0
7	2	54	7	33	0	0	0	0	3	0	1	0	0	0	3	0	0	0	0	0	0	1	3	0
8	1	0	0	0	0	0	0	0	2	2	0	0	2	0	0	17	0	3	0	0	0	3	0	0
9	3	0	0	3	0	0	0	0	0	0	0	0	0	0	2	9	0	0	1	0	0	0	4	0
الف	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	2	0	1	0	0	0	0	0	0
ب	4	0	0	0	0	0	2	0	0	0	0	0	0	0	3	0	0	0	0	11	0	1	1	0
س	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	15	0	0	0
د	0	0	0	0	1	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	2	1
ع	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
ق	0	0	0	0	7	1	0	0	0	0	0	0	0	0	0	1	0	4	0	24	0	0	1	0
هـ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
ح	0	6	0	0	0	0	1	0	0	0	0	0	5	10	0	0	0	0	0	0	0	1	0	0
ل	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	9	0	0	0	0
م	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
ن	0	0	0	0	0	0	0	0	0	0	4	0	0	0	1	0	0	2	0	0	0	0	0	0
ص	0	0	0	0	0	0	0	0	0	0	0	17	0	0	0	0	0	0	0	0	0	0	0	0
ط	1	0	0	0	0	0	0	0	0	0	2	0	0	0	2	2	0	0	0	3	0	0	0	0
و	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
ی	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Corresponding Author:
Amir Ebrahimi Ghahnavieh
Mobile Robots Research Laboratory,
Faculty of Electrical Engineering,
Amirkabir University of Technology,
Tehran, Iran
E-mail: amir.ebrahimi66@gmail.com

References

1. Y. Hu, F. Zhu, and X. Zhang. A Novel Approach for License Plate Recognition Using Subspace Projection and Probabilistic Neural Network. Vol. 3497, New York: Springer-Verlag, pp. 216–221, 2005.
2. C. Anagnostopoulos, I. Anagnostopoulos, E. Kayafas, and V. Loumos. A License Plate Recognition System for Intelligent Transportation System Applications. IEEE Trans. Intell. Transp. Syst., Vol. 7, No. 3, pp. 377–392, 2006.
3. A. Rahman, W. Badawy, and A. Radmanesh. A Real Time Vehicle's License Plate Recognition System. in Proc. IEEE Conf. Advanced Video Signal Based Surveillance, pp. 163–166, 2003.
4. K. K. Kim, K. I. Kim, J. B. Kim, and H. J. Kim. Learning-Based Approach, for License Plate Recognition. in Proc. IEEE Signal Process. Soc. Workshop, NNs Signal Process, Vol. 2, pp. 614–623, 2000.
5. Renata F. P. Neves, Alberto N. G. Lopes Filho, Carlos A. B. Mello, Cleber Zanchettin. A SVM Based Off-Line Handwritten Digit Recognizer. IEEE International Conference on Systems, Man and Cybernetics (SMC), pp. 510–515, 2011
6. A. Bellili, M. Gilloux, P. Gallinari. An MLP-SVM Combination Architecture for Offline Handwritten Digit Recognition. International Journal on Document Analysis and Recognition, Vol. 5, pp. 244–252, 2003.
7. C. Cortes and V. Vapnik. Support-Vector Network. Machine Learning, Vol. 20, pp. 273–297, 1995.
8. Javad Sadri, Ching Y. Suen, Tien D. Bui. Application of Support Vector Machines for Recognition of Handwritten Arabic/Persian Digits. 2nd Conference on Machine Vision and Image Processing, Vol. 1, pp. 300–307, 2003.
9. Tiago C. Mota, Antonio Carlos G. Thome. One-Against-All-Based Multiclass SVM Strategies Applied to Vehicle Plate Character Recognition. Proceedings of International Joint Conference on Neural Networks, Atlanta, Georgia, USA, pp. 2153–2159, 2009.
10. John C. Platt, Nello Cristianini, John Shawe-Taylor. Large Margin DAGs for Multiclass Classification. Advances in Neural Information Processing Systems 12, pp. 547–553. The MIT Press, 2000.
11. Duda R. O., Peter E. Hart, David G. Stork. Pattern Classification. 2nd edition, Wiley-Interscience, 2000
12. Bishop C. M. Pattern Recognition and Machine Learning. In M. Jordan, J. Kleinberg, B. Schölkopf editors, Springer, 2006.
13. http://www.baninick.com

10/24/2012