CAPTCHA RECOGNITION

*Pedro Henrique Lopes Silva, Filipe Eduardo Mata dos Santos**

Federal University of Ouro Preto
Department of Computing

ABSTRACT

This project is a work proposed in Pattern Recognition. The project consist in a Captcha Recognition. The captcha contains 6 characteres(digitis and uppercase letters). The system is divided in two main parts, first, segmentation, and second, the process to recognize the character. In this paper we will compare some methods to extract features and some classifiers. We achieved the 86.9369% accuracy of character recognition and 39.34% of captcha recognition in test base with the mix of HoG and SVM.

***Index Terms*—** Character Recognition, Pattern Recognition, CAPTCHA

1. INTRODUCTION

In 2000 Luis von Ahn, Manuel Blum, Nicholas Hopper and John Langford (Carnegie Mellon University) created the Completely Automated Public Turing Test To Tell Computers and Humans Apart, or just CAPTCHA.

Nowadays, we can find a several bots that do many jobs in sites automatically, but many of these sites don't want that, so the CAPTCHA was created for this, a program that protects websites against bots by generating and grading tests that humans can pass but current computer programs cannot. For that, this application uses some technique that will be presented in the next sections [1].

CAPTCHAS can be used in many applications, including:

- Free Email Services. Used to avoid "bots" that sign up for a lots of email accounts every minute;
- Search Engine Bots. Avoid indexing by search engines;
- Worms and Spam. Avoid email spam;
- Precent Dictionary Attacks. Prevent dictionary attacks in password systems;
- Preventing Comment Spam in Blogs;
- Protecting Website Registration.

Althoug we can see some pratical example, for instance, Google improves CAPTCHAS service by blocking access to automated spammers, eBay improves its marketplace by blocking bots from flooding the site with scams, and Facebook limits creation of fraudulent profiles used to spam hones users or cheat at games.

"Reverse Turing tests" is a other name for CAPTCHA, it is because they are planed to allow a computer to determine if a remote client is a other computer or a real person.

CAPTCHA is now almost a standard security technology. The most widely deployed CAPTCHAs are text-based schemes, which typically require users to solve a text recognition task. The state of the art of CAPTCHA design suggests that such text-based schemes should rely on segmentation resistance to provide security guarantee, as individual character recognition after segmentation can be solved with a high success rate by standard methods such as neural networks.

A method to recognize a CAPTCHA can be divided into five generic steps: *pre-processing*, *segmentation*, *post-segmentation*, *recognition* and *post-processing*. In pre-processing we clean the background to facilitate the next step, the segmentation, where we divided the caracteres from the others. The post-segmentation the results of segmentation are normalized (all images will have to be in the same size to increase the accuracy), the next step is the recognize each caracter and the last is a process to increase the result, where, for example, we compare the result of the last step with a dictionary.

We will work with the CAPTCHA as in the figure 1

Fig. 1. Example of CAPTCHA.

1.1. Results Obtained

It was verified that the SVM together HoG produced a good results, with accuracy 86.94% of character recognition.

*Pattern Recognition - BCC448, UFOP, October 30, 2014

1.2. Overview

This work is divided as follows, first, in section 2 we show the others works in the literature and in the next, section 3, we introduce some concepts of CAPTCHA and in the section 5 we present the idea of the tecniques. Then, in section 6 we will describe the experiments and the results. In the last section, section 7, the conclusions will be presented.

2. RELATED WORK

We have works in the literature that extract the features from the image automatically with unsupervised feature learning, we can see that in the papers [2, 3, 4]. In the paper [2], the authors apply methods recently developed in machine learning, specifically, large-scale algorithms for learning the features automatically from unlabeled data and show that can construct highly effective classifiers for both detection and recognition to be used in a high accuracy end-to-end system. A continuation of the preview work is the paper [3], from 2012, where the authors combine the representational power of large, multilayer neural networks together with recent developments in unsupervised feature learning, which allows them to use a common framework to train highly-accurate text detector and character recognizer modules. With this integration of these two modules into a full end-to-end they the state-of-the-art was achieved on standard benchmarks, namely Street View Text and ICDAR 2003. Lastly in [4], the authors attacked the problem of recognizing digits in a real application using unsupervised feature learning methods: reading house numbers from street level photos.

Although, we have works where the focus is just the segmentation like [5, 6]. In the [6], the authors proposed a character segmentation techniques of general value to attack a number of text CAPTCHAs, including the schemes designed and deployed by Microsoft, Yahoo and Google. The Microsoft CAPTCHA have been used since 2002 at many of their online services including Hotmail, MSN and Windows Live. This captcha was designed to be segmentation-resistant, however, a simple attack has achieved a segmentation success rate of higher than 90%. Already in [5] a study is done about CAPTCHAS, how can we construct a really good of it. The authors identified a series of recommendations for producing more reliable human/computer distinguishers and they proposed a framework. Applying their framework to 15 current CAPTCHA schemes from popular web sites, they find that 13 are vulnerable to automated attacks.

To facilitate the steps of segmentation and recognizing we have to papers [7] that assists the segmentation and [8] that aids the recognizing process. To facilate the process of classify we must have the most representative set of features and sometimes we can demonstrate the same features with less information, in this meaning the work [8] propose a method to decrease the size of features' vector, autoencoder. They described an effective way of initializing the weights that allows deep autoencoder networks to learn low-dimensional codes that work much better than principal components analysis(PCA) as a tool to reduce the dimensionality of data. To increase the sucess of the segmentation the paper [7] revealed the best combination of binarization methods and parameters and presents an in-depth analysis of the multimethod binarization scheme for better character segmentation. They carried out an extensive quantitative evaluation, which showed a significant improvement over conventional single-method binarization methods. Experiment results of six binarization methods and their combinations with different test images are presented.

We have the step of classification after the segmentation, in this way we have the paper [9]. In the [9], the authors classify digits of real-world house numbers using convolutional neural networks (ConvNets). Furthermore, they analyzed the benefits of different pooling methods and multi-stage features in ConvNets.

3. THEORETICAL FOUNDATION

To have a good CAPTCHA we have to follow some characteristcs.

3.1. Anti-segmentation

Segmentation is a important step of a OCR, so and itś a important characterisc of a CAPTCHA. When we have a lots of noise in the background it difficults the segmentation step, and we hold thre main ways to achieve a background confusion:

3.1.1. Background Confusion

We have three main ways to achieve this, using:

1. *Complex background*, the idea behind this is that the lines/shapes "inside it" will be confused with the real text and thus will prevent the breaker from isolating and segmenting the captcha.

2. *Color similarity*, the approach is to use colors that are perceived as very different by humans but are in reality very close in the RGB spectrum.

3. *Noise*, is the "most efficient" technique used to confuse the segmentation, itś just add random noise to the image.

3.1.2. Lines

We have two main ways to achieve this, using:

1. *Small lines*, that will prevent the captcha from being segmented.

2. *Big Lines*, the approach is to use lines that have the same ”width” as the character segments.

3.1.3. *Collapsing*

We have two main ways to achieve this, using:

1. *Predictable collapsing*, it's just remove the space between characters in the CAPTCHA.
2. *Unpredictable collapsing*, it's occurs when the number of characters is unknown average size of each character is unpredictable.

3.2. Anti-recognition

To difficult the recognition we have many ways, like we see below:

1. *Multi-fonts*, different fonts in the same CAPTCHA.
2. *Charset*, different charset the scheme uses.
3. *Font Size*, different size of characters.
4. *Distortion*, distorting the CAPTCHA globally using attractor fields.
5. *Blurring*, smudge the image.
6. *Tilting*, rotating characters with various angle.
7. *Waving*, rotating the characters in a wave fashion.

4. CLASSIFIERS

We used two classifier to compare, Support Vector Machine(SVM) and K-Nearest Neighbors (KNN).

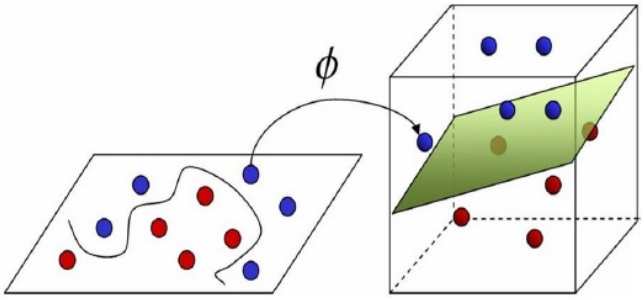
4.1. SVM

Given a set of training examples, each marked as belonging to some category, the SVM builds hyperplanes in n-dimensional space to separate each set, this can be seen in the picture 2 (a). We can see the SVM model as a representation of a set of points in space, so we divided those points comparing with what were mapped to separate categories and divided by a clear range that is as wide as possible, it may be so great as in the image 2(b) and not as 2 (c). It will classify new examples mapping within the joint created [10].

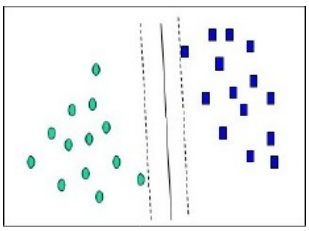
4.2. KNN

We have a point, to define the label of this we will find the k nearest neighbors, the label will be the label with most occurs in those k points. In the figure 3 with can see the operation of the KNN, for instance if we assume the k equal 3 (three) the label of the green circle it will be red triangle, because we have two triangles and just one blue square, although if we assume k equal 5 (five) the label will be blue square, because we have three blue squares and two triangles, so the label is dependent of the number of neighbors [11].

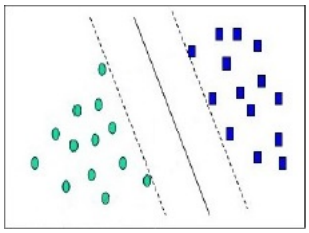
The number of neighbors influences the label what KNN will give to a point.



(a) Mapping the feature space



(b) Perfect seperation, not great



Perfect and great seperation

Fig. 2. Details SVM

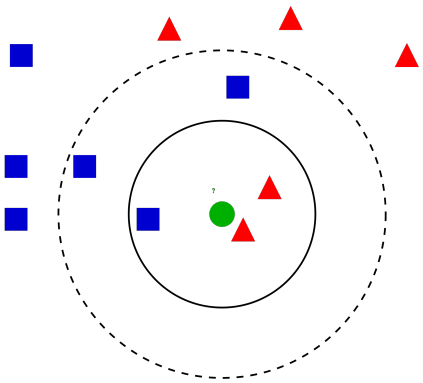


Fig. 3. Details knn

5. IMPLEMENTAÇÃO

The system is divided in two parts, the segmentation and the recognition.

5.1. Segmentation

The method to segment the captchas are simple, because the chars in the CAPTCHA are always in the same position, so we just have to take the chars always in the same position. Although, this process sometimes cut the chars and if the segmentation fail the next step, recognition, will fail too, so itĺl decrease the accuracy of the system.

5.2. Recognition

To do recognition we divided the process into two parts, features extraction and classification. The classification we use the SVM and the KNN and in the features extraction we use PCA[1], Autoencoder and HoG [2].

6. EXPERIMENTOS AND RESULTS

We did a several experiments with the features extraction Autoenconder, PCA and HoG and the classifiers SVM and KNN, however the combination HoG and SVM generated the best result with an accuracy of captcha recognition of 39.34%.

We divided the experiments dataset into three parts, training with 8004 characters (1334 captchas), validation 1998 (333 captchas) characters and test with 1998 characters (333 captchas). We divided in this way to find the best combination in the validation and after this apply the same configuration in the test base.

Extractor	**Classifier**	**ACHR(%)**[3]	**ACAR(%)**[4]
Autoencoder	SVM	7.95596	0.0
PCA	SVM	33.2833	0.0
HoG	SVM	86.9369	39.34
HoG	KNN	83.03	30.03

7. CONCLUSION

In this job we did a OCR of captchas, because they've been used for many years, and it's the motivation of this work. After all experiments we can assert that the combination of SVM and HoG result the best accuracy 86,64% of character recognition and 39,34% of captcha recognition.In contrast, SVM combination with Autoencoder was very inefficient. This combination took around 1 hour to extract the features and complete the training, and had an accuracy of only 7.96% character recognition, and of course, 0% recognition captchas. In future jobs we will use others dataset and others methods.

[1]The code can be found in http://www.dr-toolbox.com/

[2]The code can be found in http://www.vlfeat.org/

[3]Accuracy of Character Recognition

[4]Accuracy of Captcha Recognition

8. REFERENCES

[1] Luis Von Ahn, Manuel Blum, Nicholas J Hopper, and John Langford, "Captcha: Using hard ai problems for security," pp. 294–311, 2003.

[2] Adam Coates, Blake Carpenter, Carl Case, Sanjeev Satheesh, Bipin Suresh, Tao Wang, David J Wu, and Andrew Y Ng, "Text detection and character recognition in scene images with unsupervised feature learning," pp. 440–445, 2011.

[3] Tao Wang, David J Wu, Adam Coates, and Andrew Y Ng, "End-to-end text recognition with convolutional neural networks," pp. 3304–3308, 2012.

[4] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng, "Reading digits in natural images with unsupervised feature learning," vol. 2011, pp. 4, 2011.

[5] Elie Bursztein, Matthieu Martin, and John Mitchell, "Text-based captcha strengths and weaknesses," pp. 125–138, 2011.

[6] Jeff Yan and Ahmad Salah El Ahmad, "A low-cost attack on a microsoft captcha," pp. 543–554, 2008.

[7] Youngwoo Yoon, Kyu-Dae Ban, Hosub Yoon, Jaeyeon Lee, and Jaehong Kim, "Best combination of binarization methods for license plate character segmentation," *ETRI Journal*, vol. 35, no. 3, pp. 491–500, 2013.

[8] Geoffrey E Hinton and Ruslan R Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[9] Pierre Sermanet, Soumith Chintala, and Yann LeCun, "Convolutional neural networks applied to house numbers digit classification," pp. 3288–3291, 2012.

[10] Ana Carolina Lorena and André CPLF de Carvalho, "Uma introdução às support vector machines," *Revista de Informática Teórica e Aplicada*, vol. 14, no. 2, pp. 43–67, 2007.

[11] László Kozma, "k nearest neighbors algorithm (knn)," 2008.