Automated CAPTCHA Solving: An Empirical Comparison of Selected Techniques

M. Korakakis, E. Magkos and Ph. Mylonas
Ionian University, Department of Informatics
Plateia Tsirigoti 7, Corfu Greece
{p12kora, emagos, fmylonas}@ionio.gr

***Abstract*—CAPTCHAs exploit the gap in the ability between a human and a machine to understand the semantics of specific multimedia content, with vast applications in computer security. In this paper we compare two techniques in automated CAPTCHA solving for text-based CAPTCHA schemes, *i.e.*, classification based on the Vector Space Model (VSM) versus a popular Optical Character Recognition (OCR) engine. For each technique, we build a CAPTCHA solver and give it specific sets of text-based challenges to break. From our results we draw conclusions whether it is efficient to create a CAPTCHA solver by applying parts of the VSM theory and implementing a Vector Space Image Recognizer (VSIR).**

Keywords—CAPTCHA; Image recognition; Semantic context extraction; VSM; OCR

I. INTRODUCTION

A *Completely Automated Public Turing Test to Tell Computers and Humans Apart* (CAPTCHA) is a program that can generate visual or audio content challenges that a human is able to pass most of the times, while a computer program is not, with non-negligible probability [1, 2]. This *semantic gap* [3], in recognizing multimedia content, between a human and a program has shown to have many applications in computer and network security: Mainly, to ensure integrity in online polls, prevent/deter worms, spam, dictionary attacks, search engine bots, denial of service (DoS) attacks, etc. [4].

Among the various types of CAPTCHA architectures [5, 6], text-based schemes are the most widely used and highly acceptable CAPTCHA form. Such schemes use a visual image containing alphabets and numbers in a text string that the user must identify and type in a text box provided near the CAPTCHA image. Typically, the challenge-image is of low quality with different forms of noise and strong degradation applied to it.

Since the creation of CAPTCHA there has been a continuous arms race between CAPTCHA designers and CAPTCHA solvers, paving the way for research into new improved and safer designs imperative. Essentially all commercial text-based CAPTCHAs have been defeated, using object-recognition techniques, with high percentages of accuracy, e.g., [7, 8, 9, 10].

Semantics-driven indexing and retrieval of multimedia content is an integral part of the automated CAPTCHA solving process. It generally involves the use of a semantic cue as a basis for the creation of a training set, the classification of the input and the acquisition of the desired information [11, 12]. In our case the use of semantic-indexing is attained by having a Vector Space Image Recognizer (VSIR) associate the input with possible related content in its corpus, construct an output and then provide a correlation between the input and the expected outcome using a number representation, *e.g.*, 0 may denote no correlation, whereas 1 may denote that we have an output, which reflects correctly the human choice.

Our Contribution. In this paper, we argue whether it is efficient to create an image recognizer based on the Vector Space Model (VSM) that is able to solve specific text-based CAPTCHA challenges. This was determined by having a VSIR compete against an already well-established technique in automated CAPTCHA solving, namely an Optical Character Recognition (OCR) engine. In contrast with the current state-of-the-art strategies in CAPTCHA solving our VSM-based approach takes advantage of the structured information that an image presents at its core. Consequently the VSIR achieves to extract the semantic context and to solve the text-based CAPTCHA challenge with success. This fundamental flaw, which the VSM uses as a basis for its functionality and can be identified in every OCR-based CAPTCHA scheme, stresses the importance of our research towards gaining a more complete understanding of the security vulnerabilities presented in current CAPTCHAs.

The remaining text of this paper is organized as follows: Section 2 presents a brief review of existing strategies and techniques in CAPTCHA solving. Section 3 provides a technical assessment of the engines that were used and illustrates the steps that were followed to compare the two suggested techniques. In Section 4 we summarize the results and lessons that were obtained from the analysis, declare certain necessary assumptions that were made during the development and the comparison phases, and suggest improvements and possible alterations that can be applied to the VSIR. Section 5 concludes the paper.

II. RELATED WORK

Solving the text-based challenge by devising an OCR-strategy. A popular strategy in the research literature is the one followed by [19, 20, 21, 14, 18], where authors attempt to solve specific text-based challenges by introducing a

978-1-4799-6814-5/14 $31.00 © 2014 IEEE
DOI 10.1109/SMAP.2014.29

combination of techniques for the transformation of the image, character extraction and recognition for each deployed CAPTCHA scheme. Besides the noise removal and segmentation techniques, which are depended on the text-based challenge, the main difference between the aforementioned strategies is the classification method being used. In [19, 20, 21], authors create a custom recognizer with the use of Support Vector Machine (SVM). In [14, 18] an already existing OCR engine is being used instead. Among those two, the first category provides better success rates, mainly because of the depth of specialization that a SVM can offer in the classification part for a specific CAPTCHA, against a generic OCR engine that is able to recognize a larger number of text-based CAPTCHAs but with less accuracy. For example in [20] the SVM was able to recognize characters with greater than 96% accuracy. Furthermore, in [14, 18] the authors acknowledge that the accuracy of generic OCR engines can reach high success rates only with additional training.

Taking advantage of critical design flaws. On the other hand, authors in [22, 23, 8] carry out a systematic study of popular text-based CAPTCHAs. By exploiting critical design errors in each CAPTCHA scheme, they create simple but novel attacks that have a high success rate. Based on their observations, authors emphasize a series of recommendations against creating flawed CAPTCHA designs. Characteristically, in [8] Yan and Ahmad managed to achieve a higher than 90% success against a scheme that was designed to be segmentation-resistant without the use of OCR techniques, by performing a simple yet efficient, in terms of computational power and success rate, attack, and declaring that this could lead to a complete crack with a greater than 60% rate.

III. TECHNIQUES COMPARISON

A. Building Blocks

The main tools that were used in this research were the Tesseract engine [13] and our own VSIR implementation, along with a provision of a corpus that contained samples of all possible characters that the decoded CAPTCHA used. Tesseract is an open-source Google-owned OCR engine that was developed at HP between 1984 and 1994. It is widely considered as one of the most accurate open-source OCR engines available [14, 13]. A VSIR is essentially an application of the VSM, where the stored entities are compared with each other or with incoming search requests. This is achieved by modelling the various information retrieval objects as elements of a vector space and by employing matrix analysis techniques to find the relations and key features in the entities [15].

B. Technical Assessment

Tesseract, through the use of multiple algorithms is able to detect proportional and non-proportional words, chop joined characters and associate broken characters without any external guidance. In addition, it uses two character classifiers, a static and a dynamic, which increases the accuracy when distinguishing upper and lower letters. One major disadvantage is that it cannot be trained against a custom dataset that contains CAPTCHA images that differ, which renders this particular OCR engine inefficient against CAPTCHAs that use multiple fonts [13].

A distinctive attribute that the VSIR possess is that it cannot become over-trained. This of course can work as a double-edged sword because increasing the corpus size can augment the accuracy but can also take a heavy toll on the classification performance, thus requiring a significant amount of processor time [24]. The VSIR cannot adapt to changes in real time. Each time a different CAPTCHA image is being added as a new element in the vector space, all existing images must be re-indexed and additional training for the new CAPTCHA is required [25]. Unlike neural networks, the VSIR needs a standard way to deal with a problem and it cannot come up with a new solution on its own.

C. Basis for Comparison

The images used for the comparison were created by the ASP.NET Security Image Generator, a CAPTCHA-generating library freely available on the Web[1]. The important characteristics to note concerning all the images that had been generated are the following: a) The individual characters are placed in equally-sized subdivisions of the image; b) The characters have a standard font; c) Background noise has been generated by placing random drawn lines across the image (Fig. 1). The first collection of CAPTCHA images contained only numbers, the second only upper-case letters and finally the last one had both upper-case letters and numbers. This was done in order to examine the efficiency of both the VSIR and Tesseract through different quantities of possible characters that they had to recognize. The first and third collections of CAPTCHAs were 6 characters long while the second, which contained only letters, did not have a specific character length due to the fact that the generator gave the option to produce only random words and not letters.

D. Noise Removal

Noise removal was crucial in order for both solvers to be able to conduct segmentation and character classification. Also it is important to note that during testing it was observed that any significant presence of noise in the picture dropped the accuracy to low levels especially for the OCR engine. The removal of the noise along with the segmentation technique, were mainly based on the steps presented in [16, 9]. The implemented algorithm initially converted the image to black and white pixels. Then it checked for multiple non-white pixels in a row and changed those pixels if their sum was less than or equal to a chopping factor that was predetermined through testing for optimal performance (Fig. 2).

Fig. 1. Image of text-based CAPTCHA produced by the ASP.NET Generator

E. Segmentation

Segmentation of characters was an important procedure for the creation of an effective training set for the VSIR. This technique was not applied in Tesseract, because it uses a dictionary to identify whole set of words, as opposed to individual characters. For that matter, feeding Tesseract with individual characters would negatively impact its accuracy

[1] http://aspsig.sourceforge.net/

[13]. The CAPTCHAs that were used did not have merged characters, so none of them were joined with each other before noise was added. This enabled the process of taking horizontal slices of the image so as to test where each character started and finished, leading to the extraction of the characters into separate images (Fig. 3). Analytical accuracy rates are depicted in Table I.

Fig. 2. Successful noise removal (*before and after*)

TABLE I. SEGMENTATION PERFOMANCE

Number of images (per CAPTCHA variation)	Percentage of failed segmentations
420 (numbers only)	7.85%
574 (letters only)	4.87%
418 (numbers and letters)	5.74%

F. Corpus Construction

A training set is an essential part of the VSIR that has a crucial influence on its efficiency. The VSIR was trained by breaking 70 different images that were created for each CAPTCHA collection into separate characters. After that, many variations of each character were organized into a corpus in order for the VSIR to be able to efficiently recognize the possible outcome of characters that the CAPTCHA contains. The first training set had an average of 64.4 character variations. The second set had 18.1 variations per character and finally the last one had 16.3 variations for each character.

IV. RESULTS AND LESSONS LEARNED

A. Success Rates

The results of the first CAPTCHA set that contained only numbers were in favour of the VSIR. Out of 50 images the VSIR was able to break correctly 31. On the other hand Tesseract was able to guess correctly 28. Again, in the second wave of testing, the VSIR achieved better results with 24 correct guesses versus 8 correct for Tesseract. Only in the final CAPTCHA set, Tesseract managed to achieve better results with 12 correct against 9 for the VSIR. Information considering the performance can be seen in Table II.

B. Attack Speed

Both solvers were implemented in Python and tested on a desktop computer with 2.26 GHz Intel Core 2 Duo and 4 GB RAM. The figures in Table III show that even though Tesseract was faster than the VSIR both times can be considered efficient. It is also important to note, that the VSIR performed significantly better when it had a smaller data set.

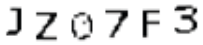

Fig. 3. Successful segmentation after noise removal

C. Assumptions

The algorithm that was constructed for this paper removes the noise and segments the characters for a specific variation of text-based CAPTCHA. However because both the VSIR and the Tesseract take as input images that do not have any background noise we can assert that as long as there is a common technique that removes correctly the noise and segments the characters with success and the VSIR has a proper training set, the results will always give VSIR the lead for any given text-based CAPTCHA. It is also important to note that even though the segmentation technique managed in average to segment the characters with 89.39% accuracy, the images that were included in the test sample were checked to ensure that they had been successfully segmented in order to prevent interference with the final results.

TABLE II. SUCCESS RATES

Size of possible CAPTCHA characters	OCR accuracy	VSIR accuracy
10 (only numbers)	56.0%	62.0%
26 (only letters)	16.0%	48.0%
36 (both numbers and letters)	24.0%	18.0 %

D. Lessons Learned and Possible Improvements

Both solvers achieved to break the CAPTCHAs successfully, considering that the reCAPTCHA developers ask that computers can solve at most 5% of the generated puzzles, or else the CAPTCHA system should be considered broken [17]. Furthermore, even with low success rates the attack can still be considered effective in the case that the attacker augments his computational power, e.g., with the use of botnets [18]. Also it is important to stress that even though the VSIR had a relatively small sample as a training set, it had overall better results against an experienced OCR engine such as Tesseract.

Important optimizations for consideration are firstly the automation of the process of creating a training corpus for the VSIR and secondly the creation of a more independent environment for the VSIR in terms of the type of image that it can take as input, through the creation of a more efficient and universal noise removal and segmentation technique. In future work, we intend to research on finding possible correlations concerning the VSIR recognition rate and time performance when the corpus has different size values and contains different samples, as well as on examining possible applications of the VSM in other CAPTCHA variations such as the image-based scheme.

TABLE III. TIME PERFORMANCE (SECONDS PER CAPTCHA SET)

CAPTCHA variation	Tesseract	VSIR
Numbers	13.62	34.35
Letters	13.50	30.80
Numbers and Letters	13.15	18.59

V. CONCLUSIONS

In this position paper, we attempted to conduct a first brief investigation of current state-of-the-art techniques regarding automated CAPTCHA solving, focusing on the application of the VSM theory towards the implementation of an efficient Vector Space Image Recognizer (VSIR). In this manner, it was demonstrated herein that the VSIR can be considered as an effective way to break CAPTCHAs. Our intention was to assess the proposed techniques in a way that sheds some light on the work in the field. As a result, this research aimed mainly in the improvement of text-based CAPTCHAs through the examination of solving techniques that can be implemented with the intention of defeating them. Among our future work is the extension of this applied technique to other CAPTCHA application domains.

REFERENCES

[1] L. von Ahn, M. Blum and J. Langford. "Telling Humans and Computer Apart Automatically", CACM, vol. 47, 2004.

[2] M. Naor, "Verification of a human in the loop or Identification via the Turing Test," On the web http://www. wisdom. weizmann. ac. il/~ naor/PAPERS/human abs. html, unpublished draft, 1996.

[3] A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta and R. Jain, "Content-based image retrieval at the end of the early years," Pattern Analysis and Machine Intelligence, IEEE Transactions on, 22(12), 1349-1380, 2000.

[4] L. von Ahn, M. Blum, N. J. Hopper and J. Langford, "CAPTCHA: Using hard AI problems for security," In: Proceedings of the 22nd international conference on Theory and applications of cryptographic techniques, EUROCRYPT 2003, pp. 294-311.

[5] M. T. Banday and N.A. Shah, "Image flip CAPTCHA," ISC International Journal of Information Security (ISeCure), pp. 105-123, July 2009.

[6] B. Jeng, C. C. Tseng, D. F. Tseng and J. C. Wang, "A study of CAPTCHA and its application to user authentication," Computational Collective Intelligence. Technologies and Applications Lecture Notes in Computer Science, vol. 6422, pp 433-440, 2010.

[7] G. Moy, N. Jones, C. Harkless, and R. Potter, "Distortion estimation techniques in solving visual CAPTCHAs," In: Proceedings of the 2004 IEEE Computer Society Conference, vol. 2, pp. 23–28.

[8] J. Yan and A. S. El Ahmad, "A low-cost attack on a Microsoft CAPTCHA," In: Proceedings of 15th ACM conference on Computer and communications security, pp. 543-554, 2008.

[9] A. A. Chandavale, A. M. Sapkal and R. M. Jalnekar, "Algorithm to break visual CAPTCHA," ICETET 2009 Proceedings of the 2009 Second International Conference on Emerging Trends in Engineering and Technology, pp. 258-262.

[10] I. J. Goodfellow, Y. Bulatov, J. Ibarz, S. Arnoud and V. Shet, "Multi-digit number recognition from street view imagery using deep convolutional neural networks," On the web at http://arxiv.org/pdf/1312.6082v4.pdf, unpublished draft, April 2014.

[11] R. Datta, J. Li and J. Z. Wang, "Exploiting the human-machine gap in image," Transactions on Information Forensics and Security, vol. 4, pp. 504-518, September 2009.

[12] F. Monay and D. Gatica-Perez, "Modeling semantic aspects for cross-media image indexing," IEEE Transactions on Pattern Analysis and Machine Intelligence , vol. 29, pp. 1082-1817, October 2007.

[13] R. Smith, "An overview of the Tesseract OCR Engine," In: Proceedings of the Ninth International Conference on Document Analysis and Recognition , vol.2, pp. 629-633, 2007.

[14] P. Baecher et al., "CAPTCHAs: The good, the bad and the ugly," In: Proceedings of Sicherheit 2010: Sicherheit, Schutz und Zuverlässigkeit, Beiträge der 5. Jahrestagung des Fachbereichs Sicherheit der Gesellschaft für Informatik e.V. (GI), 5.-7. Oktober 2010.

[15] G. Salton, A. Wong and C.S Yang, "A vector space model for automatic indexing," Communications of the ACM, vol. 18, pp. 613-620, November 1975.

[16] Priyanka, H. Kaur and D. K. Kushwaha, "Reviewing effectiveness of CAPTCHA," International Journal of Computer Trends and Technology (IJCTT), 2013.

[17] L. von Ahn, B. Maurer, C. McMillan, D. Abraham and M. Blum, "reCAPTCHA: Human-based character recognition via Web security measures," Science, vol. 321, pp. 1465–1468, Septempber 2008.

[18] J. Wilkins, "Strong CAPTCHA guidelines v1.2," On the web at http://123seminarsonly.com/Seminar-Reports/008/47584359-captcha.pdf, unpublished draft, December 2009.

[19] P. Golle, "Machine learning attacks against the Asirra CAPTCHA," In: Proceedings of the 15th ACM conference on Computer and communications security, pp. 535-542, 2008.

[20] K. Chellapila, K. Larson, P. Simard and M. Czerwinski, "Computers beat humans at single character recognition in reading-based Human Interaction Proofs," In: 2nd Conference on Email and Anti-Spam (CEAS'05), 2005.

[21] M. Wang, T. Zhang, W. Jiang and H. Song, "The recognition of CAPTCHA," Journal of Computer and Communications, vol.2, pp. 14-19, January 2014.

[22] E. Bursztein, M. Martin and J. C. Mitchell, "Text-based CAPTCHA strengths and weaknesses," In: Proceedings of the 18th ACM conference on Computer and communications security, 2011.

[23] J. Yan and A. S. El Ahmad, "CAPTCHA Robustness: A security engineering perspective," Computer, vol. 44, pp. 54-60, February 2011.

[24] H. Chuang and K. Seamons, "Document ranking and the vector space model", Software IEEE, vol.14, pp. 67-75, 1997.

[25] V. V. Raghavan and S. K. M. Wong, "A critical analysis of vector space model for information retrieval", Journal of the American Society for Information Science, vol.37, pp. 279-287, Semptember 1986.