Breaking the PayPal HIP: A Comparison of Classifiers

Kurt Alfred Kluever
Document and Pattern Recognition Lab
Department of Computer Science
Rochester Institute of Technology
Rochester, NY 14623 USA
kurt@kloover.com

May 20, 2008

Abstract

Human Interactive Proofs (HIPs) are a method used to differentiate between humans and machines on the internet. Providers of online services such as PayPal.com use HIPs to prevent automated signups and abuse of their services. In this experiment, a three step algorithm has been developed to break the PayPal.com HIP. The image is *preprocessed* to remove noise using thresholding and a simple cleaning technique, and then *segmented* using vertical projections and candidate split positions. Four *classification* methods have been implemented: pixel counting, vertical projections, horizontal projections and template correlations. The system was trained on a sample of twenty PayPal.com HIPs to create thirty-six training templates (one for each character: 0-9 and A-Z). A sample of 100 PayPal.com HIPs were used for testing. The following HIP success rates have been achieved using the different classifiers: 8% pixel counting, vertical projections 97%, horizontal projections 100%, template correlations 100%. Three of the classifiers out perform the 88% HIP success rate of [6].

1 Introduction

Human Interactive Proofs (HIPs) are a method used to automatically differentiate between humans and machines on the internet. HIPs should be easy for a machine to automatically generate, easy for a human to solve, and difficult, or impossible, for a machine to solve. They are typically implemented as an image of distorted text which the user must correctly transcribe. However, nearly all of the text recognition based HIPs are insecure against attacks based on neural networks [3], shape matching [11], and simple pattern recognition [17]. In this experiment, the PayPal.com HIP has been successfully broken using the conventional OCR process: *pre-processing, segment, classify.* Four

sults are compared using the following metrics: *HIP accuracy*, *character accuracy*, *confidence*, and *running time*. The classifiers are based on the following methods: pixel counting, vertical projections, horizontal projections, and template correlations. Section 2 provides a background on OCR and HIPs. The experiment's methodology is explained in Section 3. Section 4 details the training stage. Section 5 explains and compares the results of the system. Section 6 summarizes the approaches and the results of the work presented.

2 Background [1]

2.1 OCR

Optical Character Recognition (OCR) is the process of translating images of handwritten, typewritten, or printed text into a format understood by machines for the purpose of editing, indexing, searching, or compression [14]. The OCR process can be broken down into three tasks: pre-processing, segmentation, and classification.

2.1.1 Pre-processing

Pre-processing is necessary if the input image is noisy due to old paper, poor printers, bad scanners, etc. Generally, the pre-processing task consists of noise removal, skew correction, and thresholding. Noise removal can be achieved through filtering the image to remove extraneous or stray marks. Skew correction can be performed by estimating the angle of the text. Thresholding is the process of setting all intensity values greater than some threshold value to "on" and is often used as a method of binarizing an image. Thresholding is often used to remove noise when the salient information has either very low or very high intensity values. These techniques attempt to provide clean (or as clean as possible) input to the segmenter.

2.1.2 Segmentation

Segmentation is the process of breaking the input image into segments which contain a single entity. Character-based segmentation is the decomposition of an image into subimages which only contain a single character. Segmentation is dependent on local decisions with regards to shape similarity, and sometimes global decisions with regards to surrounding context. It is a critical step in most OCR systems, and typically the cause of a high proportion of OCR errors. In 1996, Richard Casey and Eric Lecolinet surveyed the available methods and defined three categories for offline character segmentation methods based on how segmentation and classification interact in the OCR process [2]:

- Dissection Approach: a single partitioning of the image into subimages based on "character-like" properties, followed by classification of the subimages
- Classification-based Approach: segmentation where the image is iteratively searched for components that most closely match the classes in the alphabet
- Holistic Approach: recognize words as single units (no character segmentation)

The task of segmenting characters varies in

pitch machine printed text can typically be

[1] Some of the contents of the background section is adapted from the author's previous research and writings performed during thesis prep.

segmented fairly easily using simple projection analysis (we have done so in this paper). In order to achieve high accuracy on complex problem domains, segmentation and recognition cannot be treated independently. However, a simple problem domain can use more basic techniques and still achieve high accuracy.

Dissection is the decomposition of the input image into a sequence of subimages. The criterion for good segmentation using the dissection approach is the agreement of character properties in the segmented subimage and the expected symbol. The character properties include height, width, separation from neighboring components, disposition along the baseline, etc. Interaction with the classifier is limited to reprocessing of ambiguous recognition results. For example, if the classifier can't make any decision at all, the segment may need to be re-split into two new segments.

The earliest and simplest form of dissection relies on vertical whitespace between successive characters. To make segmentation even easier, a fixed pitch if often used (*pitch* is the number of characters per unit of horizontal distance). In applications that use machines to print the text, text is often printed with a

and Mccullough [7] designed a system that could aid in segmentation when a fixed pitch could not be enforced. The system consisted of three steps: 1) detection of the start of a character based on an a priori pitch measurement, 2) a decision to begin testing for the end of the character, and 3) detection of the end of a character. The authors reported a 97% accuracy, but the results were heavily dependent on the quality of the input image.

Projection analysis is another simple one-dimensional segmentation method that uses the vertical projection (or vertical histograms) of "on" pixels to determine dissection candidates. A vertical projection is simply a column-wise count of "on" pixels. If the count falls below a predefined threshold, the column is a candidate for splitting the image. The segmentation boundaries can be further emphasized by observing the derivative of the vertical projection data (well-defined peaks will occur at the character boundaries). Projection analysis works well on high quality machine printed documents. However, vertical projection performs poorly on text which is italicized machine printed text, handwritten text (naturally slanted), or has not been properly skew-corrected.

In graph theory, a connected component is a maximal connected subgraph where two vertices belong to the same connected component if and only if there is a path between them. Connected component analysis can be applied to character segmentation by viewing the character's pixels as block or line adjacency graphs. Intuitively, a character is a single connected component because all pixels "touch" each other (with the exceptions of i's and j's). Connected component analysis is a two-dimensional analysis that works well on proportional fonts and handwritten characters. Connected component analysis works by labeling connected areas of black pixels as components. The components are further processed by drawing bounding boxes around them or based on a detailed analysis

determine the maximum or minimum size of the bounding boxes. Unfortunately, if characters are broken into multiple pieces (due to pre-processing artifacts, noise, etc.), connected component analysis yields poor segmentation results.

Classification-based segmentation bypasses the requirement to discretely segment the word. No complex dissection algorithm is required. Instead, the segmenter interacts directly with the classifier. Without regard to content, a mobile variable-width window blindly divides the image into many overlapping pieces and chooses the correct segmentation based on the classifier's confidence of the sampled window. Therefore, the criterion for good segmentation is the classification confidence given by the classifier of the subimage.

If the words to be recognized are dictionary words, N-gram statistics can be introduced to classification-based approaches to prune the search space [8]. For example, assuming we have already recognized the letters 't' and 'h', there is a higher probability that the next letter is an 'e' instead of a 'c'. After narrowing the possible guesses, a dictionary can be used to eliminate incorrectly spelled words in favor of correctly spelled words.

Holistic approaches attempt to recognize entire words as single units. The criterion for good segmentation are same as the criterion for good dissection, but using words as the alphabet instead of individual symbols or characters. Unlike the previously mentioned methods, a holistic approach requires a predefined lexicon. For many applications, such as check recognition or postal code reading, this constraint is satisfiable.

2.1.3 Character classification

Character classification is strongly dependent on feature vectors which are extracted from the characters. Feature extraction is the process of transforming the input data into a reduced representation. It is commonly employed when their is too much input data to efficiently process or if the input data is redundant (lots of *data* but not much *information*). This simplification of the input image provides an accurate description of a larger set of data. A common feature vector is image projections which represent the character as a vector of projection counts (discussed above). Naive methods feed entire image matrices to the classifier, while others require experts to develop visual cues to distinguish characters from one another [13]. However, if no such experts exist, other dimensionality reductions, such as *Principle Component Analysis* (PCA), can still be performed. PCA is used to reduce the dimensionality of multidimensional data sets by removing characteristics about the data which have low impact on the variance of the overall dataset. Similarly, it attempts to retain characteristics which contribute most to its variance.

Once the feature vectors are computed,

tion can be done by finding the nearest neighbor, neural networks [10, 1, 13], or other techniques. However, the classification method is largely non-important in the recognition process; by far the most important decision is the selection of features.

2.2 HIPs

Human Interactive Proofs (HIPs) are a class

ate between legitimate human users and automated, malicious robots on the internet. HIPs have many practical security applications, including preventing the abuse of online services such as free email providers. The term HIP is preferred over the more common (and unfortunately trademarked) term, *Completely Automated Public Turing tests to tell Computers and Humans Apart* (CAPTCHAs). HIP challenges should be easy for a machine to automatically generate, easy for a human to solve, and difficult, or im-

possible, for another machine to solve. The key to developing a successful HIP challenge

problem where a gap exists between human and machine capabilities.

Researchers have suggested HIPs based on hard artificial intelligence problems such as natural language processing [16], character recognition [4], image understanding [5], and speech recognition [9]. Most commercial implementations require the user to transcribe a string of distorted characters with background noise. This type of HIP can be considered broken through techniques such as shape matching [11], distortion estimation [12], and even simple pattern recognition [17]. Unfortunately, most commercial implementations are even easier to break than the research/academic implementations (as this experiment clearly demonstrates).

3 Method

To break the PayPal.com HIP, the problem can be reduced to an OCR task. As mentioned before, the OCR task can be broken down into three steps: *pre-processing*, *segmentation*, and *classification*. For clarity sake, the code is also separated into these three distinct steps.

3.1 Pre-processing

The pre-processing step is arguably the most important step when the image contains adversarial noise (such as HIPs). The noise placed on top of the HIP challenge images

tems. Before successful segmentation or classification can occur, the noise must be removed. The first step in our process is to convert the image to greyscale. The image is then thresholded to remove the noise (horizontal and vertical lines). The background thresholding technique is incredibly simple but removes nearly all of the noise in the image. Occasionally, additional noise still remains after this step. Therefore, additional cleaning is performed to remove pixels where the entire row has very few "on" pixels. A bounding box is then placed around the string of characters and cropped out.

(a) The original HIP image.

(b) After greyscaling.

(c) After thresholding.

(d) After cleaning.

(e) After bounding.

Figure 1: The pre-processing step.

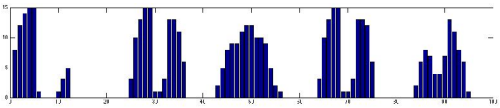
3.2 Segmentation

Next, the pre-processed image is fed into the segmenter. A connected components approach was first attempt, but unfortunately the pre-processing step occasionally breaks characters into multiple segments (see the first character in Figure 2a). However, the segmentation process is simplified because the PayPal HIP is always rendered with ex-

and candidate split positions are used to de-

termine segmentation boundaries. Splitting on every projection with zero "on" pixels occasionally causes characters to be split into multiple segments (as was the problem with the CC-based approach). However, empirical exploration shows that every character is at least ten pixels wide. A Hoffman and Mccullough style approach [7] is used and a column-wise scan is performed from the left side of the image to the right side. When the start of a character is detected, ten pixels are skipped and the scan is continued. When the end of a character is detected, the segment is cropped out of the image. The segment is padded out using 0's to a fixed size (20×20) as a requirement of the classification process (correlation requires that the dimensions of the two matrices must agree). This process is repeated until the end of the image has been reached.

(a) Input to segmenter.



(b) Vertical projection analysis.

(c) After splitting the image.

(d) After padding.

Figure 2: The segmentation step.

3.3 Classification

The segmenter feeds the five individual characters to the classifier. Note that the input images are binary images, consisting of 1's for the foreground (the character) and 0's for the background. The classification procedures (defined below) are invoked with the unknown sample image I and the set of template images T. Several of the classifiers computed correlation coefficients. The correlation coefficient (CORR2) between two input vectors or matrices i and j, can be computed as follows:

$$\frac{\sum_m \sum_n (i_{mn} \quad \bar{i})(j_{mn} \quad \bar{j})}{\sqrt{\left(\sum_m \sum_n (i_{mn} - \bar{i})^2\right)\left(\sum_m \sum_n (j_{mn} - \bar{j})^2\right)}}$$

where $\bar{i}$ is the mean of the input matrix i and $\bar{j}$ is the mean of the input matrix j.

3.3.1 Pixel Counting

The pixel counting classifier compares the Euclidean distance between pixel counts. The pixels for a binary image I can be counted using the following algorithm:

```
PIXELCOUNT(I)
1  k    0
2  for r ⇐ 1 to I_numRows
3        do for c ← 1 to I_numCols
4                 do k    k + I[r][c]
5  return k
```

The pixel count of the input image is then compared against the pixel counts of each of the template images. The index of the template that has the least difference in pixel count is returned as the match:

```
CLASSIFYPC(I, T)
1  D ← ∅
2  for each Template t_i ∈ T
3        do d_i    abs(PIXELCOUNT(t_i) -
4             PIXELCOUNT(I))
5  return k such that d_k = min(D)
```

Note that this method does not take any spatial layout into account. Therefore, an image with a pixel in each of it's four corners will match perfectly with a template image with a block of four pixels in the center of the image, even though they are visually dissimilar.

3.3.2 Vertical Projections

The vertical projection classifier compares correlation coefficients of vertical projections. The vertical projection of an image I can be calculated using the following algorithm:

VERTICALPROJECTION(I)
1 $V \leftarrow \emptyset$
2 **for** $r \leftarrow 1$ **to** $I_{numRows}$
3 **do for** $c \leftarrow 1$ **to** $I_{numCols}$
4 **do** $v_c \leftarrow v_c + I[r][c]$
5 **return** V

The vertical projection of the input image is then compared against the vertical projections of each of the template images. The index of the template whose vertical projection has the the highest correlation coefficient with the input image's vertical projection is returned as the match:

CLASSIFYVP(I, T)
1 $R \leftarrow \emptyset$
2 **for** each Template $t_i \in T$
3 **do** $r_i \leftarrow$ CORR2(
4 VERTICALPROJECTION(t_i),
5 VERTICALPROJECTION(I))
6 **return** k such that $r_k = max(R)$

3.3.3 Horizontal Projections

The horizontal projection classifier compares correlation coefficients of horizontal projections. The horizontal projection of an image I can be calculated using the following algorithm:

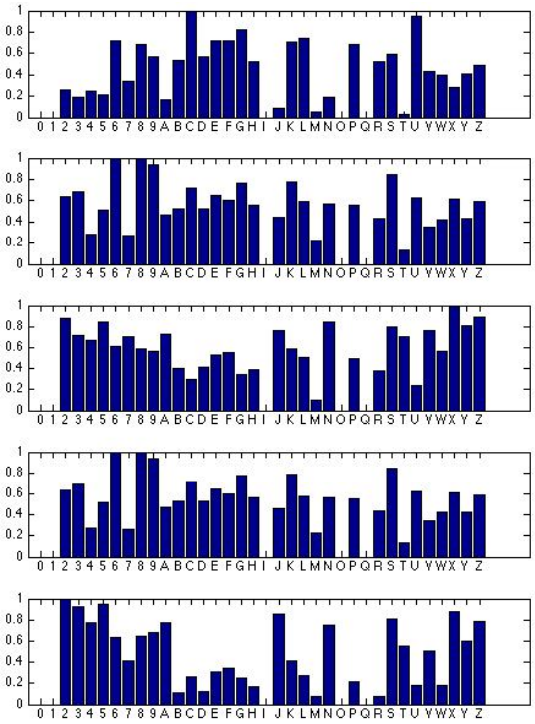


Figure 3: VP confidences for "C6X62".

HORIZONTALPROJECTION(I)
1 $V \leftarrow \emptyset$
2 **for** $r \leftarrow 1$ **to** $I_{numRows}$
3 **do for** $c \leftarrow 1$ **to** $I_{numCols}$
4 **do** $v_r \leftarrow v_r + I[r][c]$
5 **return** V

The horizontal projection of the input image is then compared against the horizontal projections of each of the template images. The index of the template whose horizontal projection has the the highest correlation coefficient with the input image's horizontal projection is returned as the match:

ClassifyHP(I, T)

```
1  R ← ∅
2  for each Template t_i ∈ T
3        do r_i ← Corr2(
4             HorizontalProjection(t_i),
5             HorizontalProjection(I))
6  return k such that r_k = max(R)
```

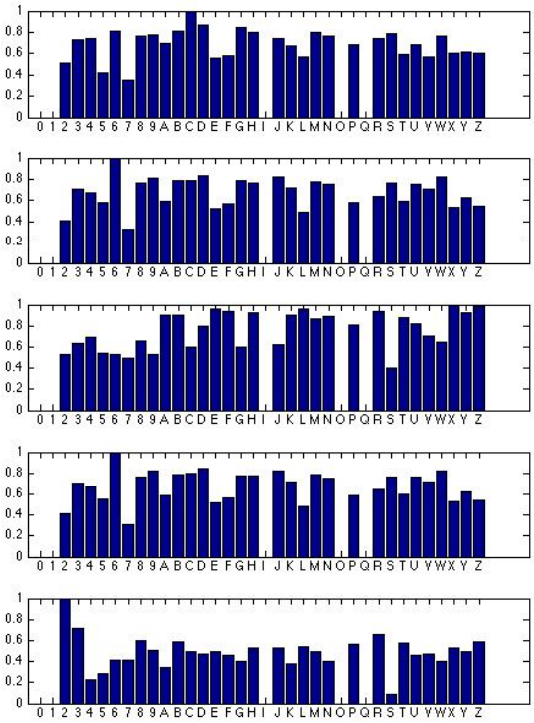


Figure 4: HP confidences for "C6X62".

3.3.4 Template Correlations

The template correlation classifier calculates

image I and the templates T. The index of the template with the highest 2D correlation coefficient with the input image is returned as the match:

ClassifyTC(I, T)

```
1  R ← ∅
2  for each Template t_i ∈ T
3        do r_i ← Corr2(t_i, I)
4  return k such that r_k = max(R)
```

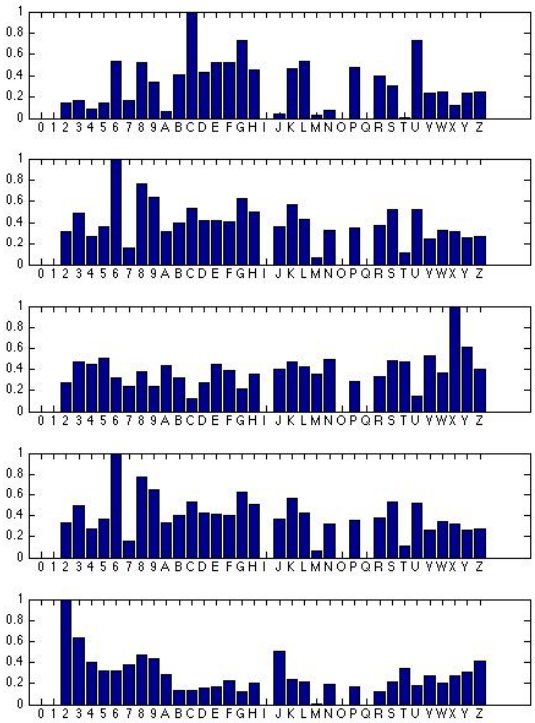


Figure 5: TC confidences for "C6X62".

4 Training

The templates were created from a set of twenty training PayPal HIPs. The set of images were randomly chosen and contain all characters in the character set (note that PayPal does not use `I`, `O`, `Q`, `0`, or `1` in the character set to increase usability for humans). The images were processed using the same pre-processing and segmentation algorithm

ing data has multiple samples for a given

character c: $s_1^c, s_2^c, \cdots, s_n^c$. If multiple samples for a single character c exist, the final template t^c is computed by averaging all samples for a given character:

$$t^c = \sum_{i=1}^{n} s_i^c \Big) / n$$

Informally, this creates more robust templates, as we are using multiple training samples to generate the templates. It can be thought of as many training samples voting on whether or not the ground truth should contain a given pixel.

5 Results

5.1 Testing Results

A sample of 100 random PayPal HIPs were used for testing. The samples were manually downloaded and labeled by visual inspection. The same pre-processing and segmentation algorithms were used for all classifiers. The classifiers are evaluated with several metrics: *HIP accuracy* refers to the percentage of the 100 testing samples which were correctly recognized. *Character accuracy* refers to the percentage of the 500 characters of the 100 testing samples which were correctly recognized. The HIP accuracy should be roughly equal to character accuracy raised to the fifth power (serial repetition). The classifiers which utilize correlation can also return a *confidence value*. The character confidence can be represented by the correlation coefficient (1.0 means a perfect match). A overall string confidence C can be calculated by multiplying each of the character correlation coefficients r_i together:

$$C = \prod_{i=1}^{5} r_i$$

Note that this confidence metric cannot be used with the pixel counting classifier because the that classifier does not utilize correlation during classification. *Running time*, measured in seconds, was clocked on a 2 GHz Intel Core 2 Duo with 2 GB of memory, running Mac OS X 10.4.11 and MATLAB R2007a. Full outputs from all four classifiers are located in Appendix B.

The following is a comparison of the four classifiers: pixel counting (PC), vertical projections (VP), horizontal projections (HP), and template correlations (TC).

	PC	VP	HP	TC
A_{Char}	63.2%	99.4%	100%	100%
A_{HIP}	8%	97%	100%	100%
C_{avg}	n/a	98.9%	98.8%	95.9%
C_{min}	n/a	89.1%	93.9%	67.9%
T_{avg}	0.02s	0.06s	0.06s	0.06s

where A_{Char} is the character accuracy, A_{HIP} is the HIP accuracy, C_{avg} is the average overall string confidence, C_{min} is the lowest overall string confidence observed during testing, and T_{avg} is the average recognition running time in seconds.

The only benefit that the pixel counting method has over the others is run

three misclassifications: '27LP5' recognized as '27LP2', '5RESL' recognized as '2RESL', and '5SMWJ' recognized as '2SMWJ'. We can see that all three mistakes were made due to a misclassification of a '5' as a '2'. This shows that vertical projections are not a sufficient method for differentiating between 5's and 2's.

Figure 6 plots the confidence of the three

horizontal projections shown in green, and template correlations shown in red) for all 100 testing samples. We can see that even

though they use different classification techniques, the confidence values seem to be consistent with one another. That is to say, that all three classifiers achieve low confidences on the same images. For example, sample #72

izontal projection and template correlation classifiers. Similarly, all three classifiers performed very well on sample #28. This indicates that samples with low confidence values may exhibit some pre-processing artifacts that make classification a difficult task, no matter what the technique.

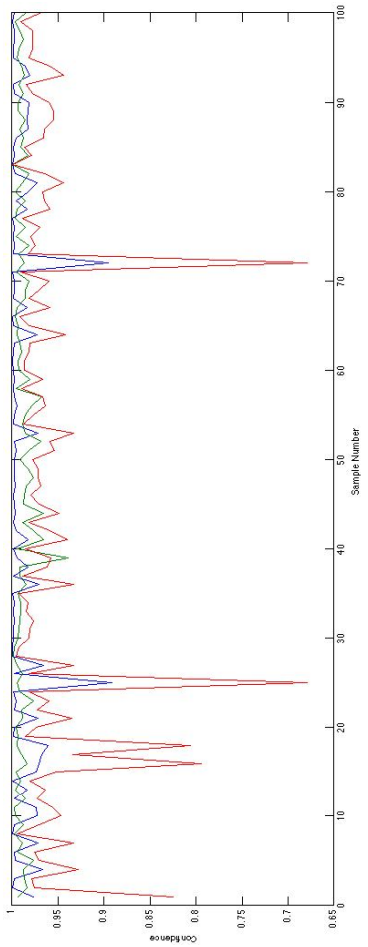


Figure 6: A comparison of confidences for the 100 test samples. VP=blue, HP=green, TC=red.

5.2 Example

This section visually demonstrates the entire recognition process using an example PayPal HIP image. Figure 1 illustrates the preprocessing step. Figure 2 displays the the segmentation process. Figure 3 shows the confidence values for the vertical projection classifier. The overall confidence for the example HIP is 0.992 (0.995 $0.999 * 1.0$ $0.998 * 1.0$). Notice that there are several other characters with very high confidences. Figure 4 shows the confidence values for the horizontal projection classifier. The overall confidence for the example HIP is 0.994 ($0.995 \quad 1.0 * 1.0 * 0.999 * 1.0$). Notice that the range of the confidence values is fairly small and several peaks exist. Figure 5 shows the confidence values for the template correlation classifier. The overall confidence for the example HIP is 0.992 (1.0 $0.999 * 0.997$ $0.996 * 1.0$). Notice that there is only a single peak in the confidence values for every character. This indicates that the classifier is very good at discriminating between character classes.

6 Conclusion

We have presented a robust way to automatically recognizing the character strings inside of a PayPal.com HIP using a three step *pre-process, segment, classify* algorithm. Four classifiers (pixel counting, vertical projections, horizontal projections, and template correlations) were implemented, evaluated, and compared. Two of the classifiers have achieved perfect HIP accuracy on the test set of 100 images. Upon visual inspection of the correlation coefficients for several test images, we see that the template correlation classifier

and is strongly recommended.

References

[1] Hadar I. Avi-Itzhak, Thanh A. Diep, and Harry Garland. High accuracy optical character recognition using neural networks with centroid dithering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(2):218–224, February 1995.

[2] Richard G. Casey and Eric Lecolinet. A survey of methods and strategies in character segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(7):690–706, July 1996.

[3] Kumar Chellapilla and Patrice Y. Simard. Using machine learning to break visual human interaction proofs (HIPs). In Lawrence K. Saul, Yair Weiss, and Léon Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 265–272, Cambridge, MA, December 2004. MIT Press.

[4] Monica Chew and Henry S. Baird. Baffletext: A human interactive proof. In *IST/SPIE Document Recognition and Retrieval X Conference*, pages 305–316, January 2003.

[5] Monica Chew and J. Doug Tygar. Image recognition captchas. In Kan Zhang and Yuliang Zheng, editors, *In Proceedings of the 7th International Information Security Conference (ISC 2004)*, volume 3225 of *Lecture Notes in Computer Science*, pages 268–279, Palo Alto, CA, September 2004. Springer Berlin / Heidelberg.

[6] Sam Hocevar. Pwntcha. Online http://libcaca.zoy.org/wiki/PWNtcha, January 2005.

[7] Richard L. Hoffman and J. Warren McCullough. Segmentation Methods for Recognition of Machine-Printed Characters. *IBM Journal of Research and Development*, 15(2):153–165, March 1971.

[8] Jr. Kenneth Crawford Hayes. Reading handwritten words using hierarchical relaxation. In *Computer Graphics and Image Processing*, volume 14, pages 344–364, December 1980.

[9] Greg Kochanski, Daniel P. Lopresti, and Chilin Shih. A reverse turing test using speech. In *Proceedings of the 7th International Conference on Spoken Language Processing*, pages 1357–1360, Denver, Colorado, September 2002.

[10] Nallasamy Mani and Bala Srinivasan.

model for optical character recognition. In *International Conference on Systems, Man and Cybernetics*, volume 3, pages 2517–2520. IEEE Computer Society, October 1997.

[11] Greg Mori and Jitendra Malik. Recognizing objects in adversarial clutter: breaking a visual captcha. In *Conference on Computer Vision and Pattern Recognition*, volume 1, pages 134–141, Madison, WI, USA, June 2003. IEEE Computer Society.

[12] Gabriel Moy, Nathan Jones, Curt Harkless, and Randall Potter. Distortion estimation techniques in solving visual captchas. In *Conference on Computer Vision and Pattern Recognition*, volume 02, pages 23–28, Los Alamitos, CA, USA, June 2004. IEEE Computer Society.