Building Segmentation Based Human-Friendly Human Interaction Proofs (HIPs)

Kumar Chellapilla, Kevin Larson, Patrice Y. Simard, and Mary Czerwinski

Microsoft Research, One Microsoft Way, Redmond, WA, USA 98052
{kumarc, kevlar, patrice, marycz}@microsoft.com

Abstract. Human interaction proofs (HIPs) have become common place on the internet due to their effectiveness in deterring automated abuse of online services intended for humans. However, there is a co-evolutionary arms race in progress and these proofs are becoming more difficult for genuine users while attackers are getting better at breaking existing HIPs. We studied various popular HIPs on the internet to understand their strength and human friendliness. To determine HIP strength, we adopted a direct approach of building computer attacks using image processing and machine learning techniques. To understand human-friendliness, a sequence of users studies were conducted to investigate HIP character recognition by humans under a variety of visual distortions and clutter commonly employed in reading-based HIPs. We found that many of the online HIPs are pure recognition tasks that can be easily broken using machine learning. The stronger HIPs tend to pose a combination of segmentation and recognition challenges. Further, the HIP user studies show that given correct segmentation, computers are much better at HIP character recognition than humans. In light of these results, we propose that segmentation-based reading challenges are the future for building stronger human-friendly HIPs. An example of such a segmentation-based HIP is presented with a preliminary assessment of its strength and human-friendliness.

1 Introduction

Human Interaction Proofs[1] (HIPs) [3] or Completed Automated Public Turing tests to tell Computers and Humans Apart (CAPTCHAs) [4] are systems that allow a computer to distinguish between another computer and a human. These systems enable the construction of automatic filters that can be used to prevent automated scripts from utilizing services intended for humans [4]. An overview of the work in this area can be found in [3]. Work on building HIPs dates back to 1997 with the first HIP being invented [13] at the DEC Systems Research Center for blocking abusive automatic submission of URLs to the AltaVista web-site (www.altavista.com). Since then numerous HIPs have been proposed and several have been adopted by companies to

[1] These are also referred to as "Human Interactive Proofs." The term "Human Interaction Proof" is preferred in this paper as it is clearer in indicating that these are tests for human interaction.

protect various services on the web. However, the basic challenge still remains the same: design a computer program that can automatically generate and grade tests that most humans can pass but current computer programs cannot pass. For a HIP to be successful in practice, it should also be fast and be capable of generating millions of unique samples a day.

Construction of HIPs of practical value is difficult because it is not sufficient to develop challenges at which humans are somewhat more successful than machines. This is because the cost of failure from using machines to solve the puzzles may be very small. In practice, if one wants to block automated scripts, a challenge at which humans are about 90% successful and machines are 1% successful, may not be sufficient, especially when the cost of failure and repetition is low for the machine [2,7,12]. At the same time, the identical challenge must not put too much burden on the human in order to avoid discouraging the use of the service. This is summarized in Figure 1. The figure shows an ideal distribution of HIPs. The sweet spot, where the HIPs are easy for humans to recognize but difficult for hackers to crack, is not guaranteed to actually exist. Furthermore, automatically generated HIPs, being random in nature, will have a distribution of difficulty, with some particular instances extending beyond the hypothesized sweet spot.

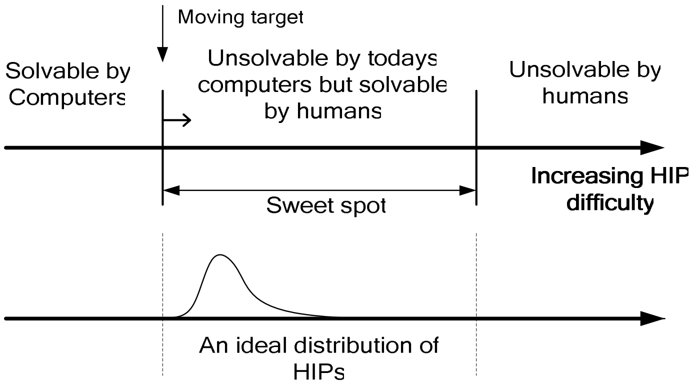

Moving target
Solvable by Computers
Unsolvable by todays computers but solvable by humans
Unsolvable by humans
Increasing HIP difficulty
Sweet spot
An ideal distribution of HIPs

Fig. 1. Regions of feasibility as a function of HIP difficulty for humans and computers algorithms.

Depending on the cost of the attack and the value of the service, automatic scripts should not be more successful than 1 in 10,000 (0.01%). For good usability the human success rate should approach 90%. While the latter is a common requirement for reducing the number of retries a human user has to endure, the former is obtained by analyzing the cost of hiring humans to solve HIPs. For example, requiring a signup HIP for creating an e-mail account only imposes a maximal cost of about .002 cents per message, while the minimum estimate for the costs/potential revenue from sending spam are around .0025 cents, with many spammers charging or earning 5 to 10 times that [12]. Thus, a practical HIP must not only be secure but also be human-friendly. Human-friendliness encompasses both a) the visual appeal and annoyance factor of a HIP, and also b) how well it utilizes the difference in ability between humans and machines at solving segmentation and recognition tasks. While HIP security considerations push designers to make the HIP difficult for both humans and computers, the human-friendliness requirements force the designer to make them only as hard as they need to be and still be effective at deterring abuse. Due to this inherent conflict between these two requirements, online HIPs are undergoing an arms race. As computer vision research advances, computers get faster, and attackers get sophisticated, existing HIPs will become ineffective and new HIPs will have to be created. Over time, the sweet spot will decrease in size. Unfortunately, humans are unlikely to get better at solving HIPs in the same timeframe [10,11].

Owing to their advantages, reading-based HIPs have become common place for protecting internet web sites against abuse. Section 2 presents motivations for a reading-based HIP and several examples of common reading-based HIPs that can be sampled on the web. It also presents the segmentation and recognition parts of the HIP challenge and key design choices that go into building a reading-based HIP. Section 3 addresses HIP security from the point of view of a computer attacker attempting to solve a HIP completely (both segmentation and recognition parts being solved) or simply the recognition part of the problem. Section 4 investigates human-friendliness of a HIP by understanding human ability in solving HIP segmentation and recognition. Section 5 reviews lessons learned from computer attacks and user studies and presents segmentation-based HIPs. A preliminary analysis of the security and human-friendliness of an example segmentation-based HIP is also presented.

2 Examples of HIPs

We have come across dozens of proposals for HIP designs, ranging from counting objects in a picture, segmenting faces, recognizing animations, identifying words in audio, etc. [4]. Among visual challenges, Reading-based HIPs are the most obvious favorite [4,5,8,12,13,14]. These HIPs are made of characters rendered to an image and distorted before being presented to the user. Solving the HIP requires identifying all characters in the correct order. Several reasons for their popularity are:

1) optical character recognition (OCR) is a well studied field and the state of the art is well known,
2) characters were designed by humans for humans and humans have been trained at the task since childhood,
3) each character has a corresponding key on the keyboard and 8 keystrokes span a space of over 1000 billion solutions,
4) localization issues are minimal using western characters and numbers (without dictionaries),
5) the task is easily understood by users without much instruction, and
6) character-based HIPs can be generated quickly[2].

Figure 2 presents reading based HIPs that can be sampled from the web while signing up for free e-mail accounts with Mailblocks (www.mailblocks.com), MSN/Hotmail (www.hotmail.com), Yahoo! (www.yahoo.com), Google (gmail.google.com), run-

[2] Over 300 8-character HIPs can be generated per second on a 3GHz P4 [2,12]

ning a whois query at Register.com (www.register.com) or searching for tickets at Ticketmaster (www.ticketmaster.com), etc.

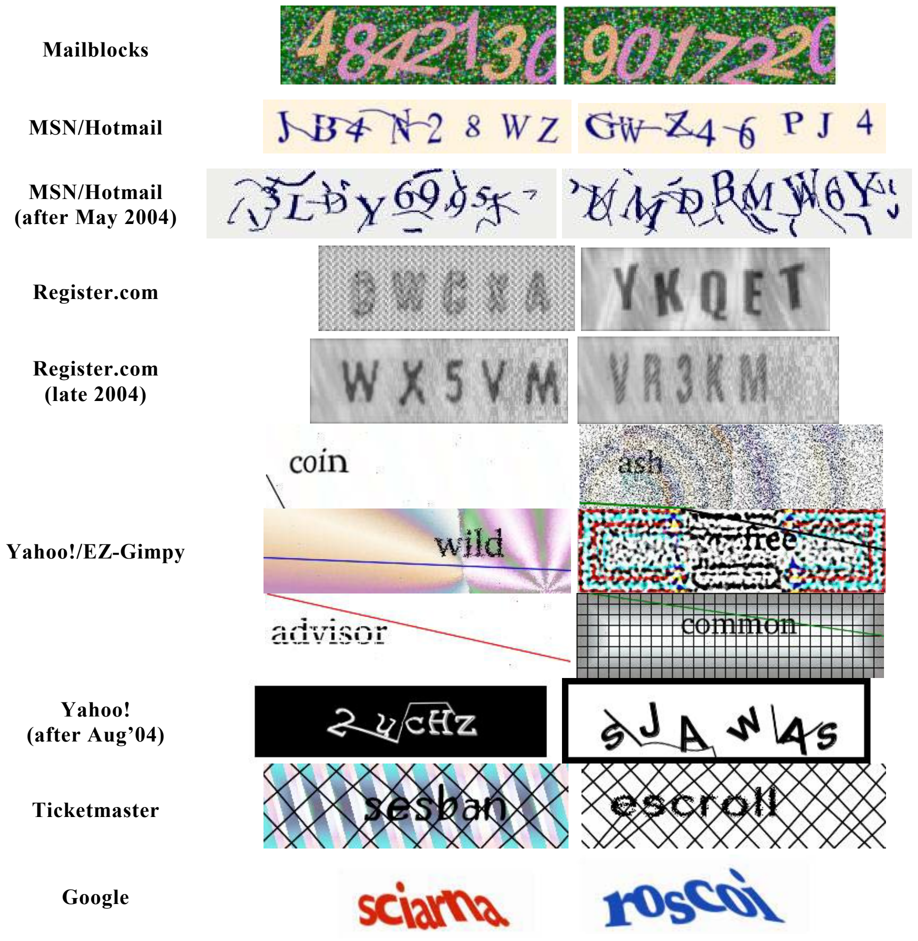


Fig. 2. Example Human Interaction Proofs (HIPs).

Solutions to Yahoo (ver1) HIPs are common English words, but those for ticketmaster and Google do not necessarily belong to the English dictionary. They appear to have been created using a phonetic generator [8]. Examining the changes in MSN, Yahoo!, and Register.com HIPs, we note that these HIPs are becoming progressively more difficult. While MSN introduced more arcs as clutter, Yahoo! gave up their language model and replaced simple textures and grids with more random intersecting lines and arcs. Register.com's update was relatively minor as they simply introduced digits into their character set.

2.1 Segmentation and Recognition Challenges

Reading-based HIP challenges typically comprise a segmentation challenge followed by recognition challenges[3]. Solving the segmentation challenge requires the identification of character locations in the right order. The random location of characters, background textures, foreground and background grids or lines, and clutter in the form of arcs make the segmentation problem difficult. Image warp exacerbates the segmentation problem by reducing the effectiveness of preprocessing stages of a segmentation algorithm that attempt to estimate and remove the background textures and foreground lines, etc. Once character locations are reliably identified (in the right order) each of the characters needs to be recognized correctly giving rise to the recognition problem. The character recognition problem is made difficult through changes in scale, rotation, local and global warp, and intersecting random arcs.

2.2 HIP Design Choices

While the segmentation and recognition challenges provide a conceptual breakdown of the HIP challenge, building an actual reading-based HIP requires one to make several independent choices:

a) **Character set**: The character set to be used in the HIP.

b) **Affine transformations**: Translation, rotation, and scaling of characters

c) **Adversarial clutter**: Random arcs, lines, or other simple geometric shapes that intersect with the characters and themselves

d) **Image warp:** elastic deformations of the HIP Image at different scales i.e., those that stretch and bend the character itself (global warp) and those that simply jiggle the character pixels (local warp)

e) **Background and foreground textures**: These textures are used to form a colored HIP image from a bi-level or grayscale HIP mask generated by using a) through d)

f) **Language model**: the language model determines both the conditional and joint probabilities of character co-occurrence in the HIP. A HIP can use a) no language model (random equally likely occurrence of all possible combinations – Eg. Mailblocks, MSN, Register and Yahoo version 2), b) words from a dictionary (Yahoo! version 1), or c) a phonetic generator [8] (Ticketmaster and Google/Gmail).

Each of these choices affects both HIP security and human-friendliness of the HIP though commonly to different degrees.

[3] Solving a HIP need not require the segmentation and recognition problems to be solved separately.

3 HIP Security

Assessing the strength of a particular HIP is an approximate process at best. The strength of a HIP is determined by the cumulative effects of the HIP design choices. Each choice increases or decreases HIP difficulty and human-friendliness. However, comparing and quantifying contributions from each choice might not be possible as interactions between these choices can be non-linear. Some very general comments can however be made. In general, the larger the character set and the longer the HIP the stronger it is. In the absence of a language model, the strength of the HIP improves exponentially with the length of the HIP and polynomially with the HIP character set size. Affine transformations, clutter, and image warp also increase HIP security through not as dramatically. Background and foreground textures usually provide only a marginal improvement in HIP security. Using only words from a dictionary makes the HIP considerably easier to break. HIPs using phonetic generators also suffer from this drawback but to a lesser extent. The effects of using a dictionary or a phonetic generator are similar to reducing the effective character set size and HIP solution length.

One direct approach to obtaining a quantitative upper bound for HIP security is to build automated HIP breakers and assess their success in solving particular HIPs. This is exactly the approach adopted here to assess HIP security for popular on-line HIPs [2,12].

3.1 Breaking HIPs

Breaking HIPs is not new. Mori and Malik [7] have successfully broken the EZ-Gimpy (92% success) and Gimpy (33% success) HIPs from CMU. Thayananthan et al [15] have also been successful at breaking EZ-Gimpy [4]. Recently Moy et al [16] have broken the purely distortion based HIP gimpy-r [4] with a success rate of 78%. Our approach aims at an automatic process for solving multiple HIPs with minimum human intervention, using machine learning. In this section, our main goal is to learn more about the common strengths and weaknesses of these HIPs rather than to prove that we can break any one HIP in particular with the highest possible success rate. We summarize results for six different HIPs: EZ-Gimpy/Yahoo, Yahoo v2, Mailblocks, Register, Ticketmaster, and Google. Further details on these HIP breakers can be found in [2,12].

To simplify our study, we will not be using language models in our attempt to break HIPs. For example, there are only about 561 words in the EZ-Gimpy dictionary [7], which means that a random guess attack would get a success rate of 1 in 561 (more than enough to break the HIP, i.e., greater than 0.01% success).

Our generic method for breaking all of these HIPs is to build a custom segmentation algorithm (to locate the characters) and then use machine learning for recognition. Surprisingly, segmentation, or finding the characters, is simple for many HIPs, which makes the process of breaking the HIP particularly easy. Gimpy uses a single constant predictable color (black) for letters even though the background color changes. We quickly realized that once the segmentation problem is solved, solving the HIP becomes a pure recognition problem, and it can be solved using machine learning. Our recognition engine is based on convolutional neural networks [6,9]. It yielded a 0.4% error rate on the MNIST database, uses little memory, and is very fast for recognition. Speed is important for breaking HIPs since it reduces the cost of automatic trials.

For each HIP, we have a segmentation step, followed by a recognition step. It should be stressed that we are not trying to solve every HIP of a given type, i.e., our goal is not 100% success rate, but something efficient that can achieve much better than 0.01%.

In each of the following HIP security experiments, 2500 HIPs were hand labeled and used as follows (a) recognition (1600 for training, 200 for validation, and 200 for testing), and (b) segmentation (500 for testing segmentation). For each of the five HIP types, a convolution neural network was trained and tested on gray level character images centered on the guessed character positions (see below). The convolutional neural network is identical to the one described in [6] and consisted of two layers of convolutional nodes followed by two fully connected layers. The output from each convolutional layer was subsampled by two before being fed to the next layer [6]. The architecture was exactly the same for all experiments in this paper. The trained neural network became the recognizer. Except for converting the original image to gray, **no preprocessing of any kind was used for recognition**.

Mailblocks: To solve the HIP, we select the red channel, binarize and erode it, extract the largest connected components (CCs), and breakup CCs that are too large into two or three adjacent CCs. Further, vertically overlapping half character sized CCs are merged. The resulting rough segmentation works most of the time. One example is presented in Figure 3.

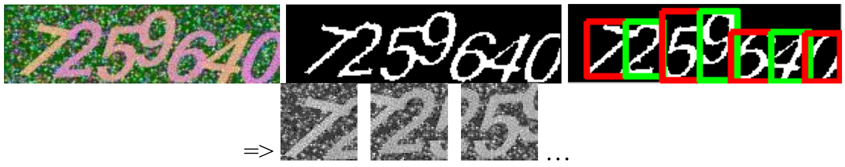


Fig. 3. Breaking Mailblocks HIP.

In the example above, the neural network would be trained, and tested on the segmented chars (right). Each HIP has exactly 7 characters. The segmentation algorithm had a success rate of 88.8% and the neural network recognition rate was 95.9% for recognition (given correct segmentation). The total end-to-end success rate is given by `seg_rate*(reco_rate)^(hip_length)` = $(0.888)*(0.959)^7$ = 66.2% total. Note that most of the errors come from segmentation, even though this is where all the custom programming was invested.

Register: The procedure to solve HIPs is very similar. The image was smoothed, binarized, and the largest 5 connected components were identified. The success rate is 95.4% for segmentation, 87.1% for recognition (given correct segmentation), and $(0.954)*(0.871)^5$ = 47.8% total. One example is presented in Figure 4.

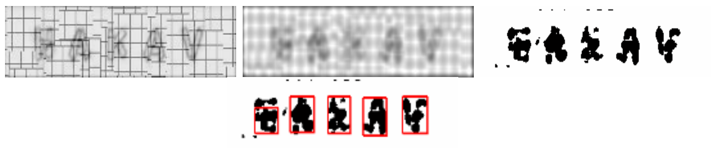


Fig. 4. Breaking Register HIP.

Yahoo!/EZ-Gimpy: Unlike the mailblocks and register HIPs, the Yahoo/EZ-Gimpy HIPs are richer in that a variety of backgrounds and clutter are possible. Though some amount of text warping is present, the text color, size, and font have low variability. Three simple segmentation algorithms were designed with associated rules to identify which algorithm to use. The goal was to keep these simple yet effective:

a) **No mesh**: Convert to grayscale image, threshold to black and white, select large CCs with sizes close to HIP char sizes. Figure 5 shows one example.

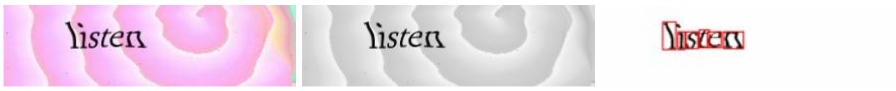


Fig. 5. Breaking Yahoo HIP: no mesh case.

b) **Black mesh**: Convert to grayscale image, threshold to black and white, remove vertical and horizontal line pixels that don't have neighboring pixels, select large CCs with sizes close to HIP char sizes. Figure 6 shows one example.

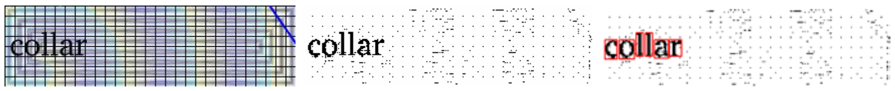


Fig. 6. Breaking Yahoo HIP: black mesh case.

c) **White mesh**: Convert to grayscale image, threshold to black and white, add black pixels (in white line locations) if there exist neighboring pixels, select large CCs with sizes close to HIP char sizes. Figure 7 shows one example.

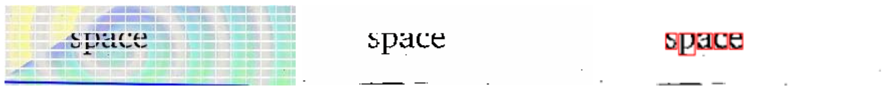


Fig. 7. Breaking Yahoo HIP: black mesh case.

Tests for black and white meshes were performed to determine which segmentation algorithm to use. The average length of a Yahoo HIP solution is 4.8 characters. The end-to-end success rate was 56.2% for segmentation (38.2% came from a), 11.8% from b), and 6.2% from c), 90.3% for recognition (given correct segmentation), and $(0.562)*(0.903)^{4.8}$ = 34.4% total. Note that the same recognizer was used for all 3 scenarios.

Ticketmaster: The procedure that solved the Yahoo HIP is fairly successful at solving some of the ticket master HIPs. These HIPs are characterized by cris-crossing lines at random angles clustered around 0, 45, 90, and 135 degrees. A multipronged attack as in the Yahoo case (section 3.3) has potential. In the interests of simplicity, a single attack was developed: Convert to grayscale, threshold to black and white, up-sample image, dilate first then erode, select large CCs with sizes close to HIP char sizes. One example is presented in Figure 8.

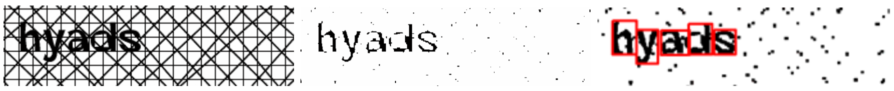


Fig. 8. Breaking Ticketmaster HIP.

The dilate-erode combination causes the lines to be removed (along with any thin objects) but retains solid thick characters. This single attack is successful in achieving an end-to-end success rate of 16.6% for segmentation, the recognition rate was 82.3% (in spite of interfering lines), and $(0.166)*(0.823)^{6.23}$ = 4.9% total. The average HIP solution length is 6.23 characters.

Yahoo! (version 2): The second generation HIP from Yahoo had several changes: a) it did not use words from a dictionary or even use a phonetic generator, b) it uses only black and white colors, c) uses both letters and digits, and d) uses connected lines and arcs as clutter. The HIP is somewhat similar to the MSN/Passport HIP which does not use a dictionary, uses two colors, uses letters and digits, and background and foreground arcs as clutter. Unlike the MSN/Passport HIP, several different fonts are used. A single segmentation attack was developed: Remove the 6 pixel border, up-sample, dilate first then erode, select large CCs with sizes close to HIP character sizes. The attack is practically identical to that used for the ticketmaster HIP with different preprocessing stages and slightly modified parameters. Figure 9 shows an example.

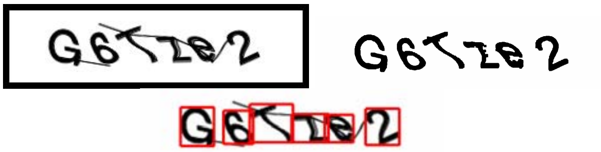


Fig. 9. Breaking Yahoo! (version 2) HIP.

This single attack is successful in achieving an end-to-end success rate of 58.4% for segmentation, the recognition rate was 95.2%, and $(0.584)*(0.952)^{5}$ = 45.7% total. The average HIP solution length is 5 characters.

Google/Gmail: The Google HIP is unique in that it uses only image warp as a means of distorting the characters. Similar to the MSN/Passport and Yahoo version 2 HIPs, it is also two colors. The HIP characters are arranged close to one another (they often touch) and follow a curved baseline. The following very simple attack was used to segment Google HIPs: Convert to grayscale, up-sample, threshold and separate connected components.

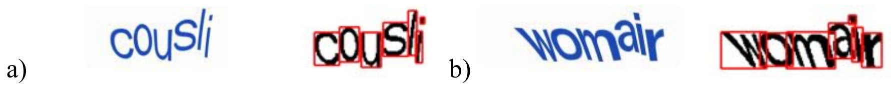


Fig. 10. Breaking Google HIP.

This very simple attack gives an end-to-end success rate of 10.2% for segmentation, the recognition rate was 89.3%, giving $(0.102)*(0.893)^{6.5}$ = 4.89% total probability of breaking a HIP. Average Google HIP solution length is 6.5 characters. This can be significantly improved upon by judicious use of dilate-erode attack. A direct application doesn't do as well as it did on the ticketmaster and yahoo HIPs (because of the shear and warp of the baseline of the word). More successful and complicated attacks might estimate and counter the shear and warp of the baseline to achieve better success rates.

The above experiments show that using a very simple approach, even the strongest HIPs can be solved more often than 1 in 25 leaving us far from the 1 in 10,000 mark. While recognition success rates ranged from 82.3% (ticketmaster) to 95.9% (mailblocks), segmentation success rates ranged from 10.2% (Google) to 95.4% (Register.com). Clearly, the segmentation problem is crucial in determining HIP security, while recognition of HIPs characters appears to be a solved problem.

3.2 Single Character Recognition Using Machine Learning

Interestingly, the recognition problems posed by the HIPs in Section 3.1 were specifically designed to fool off-the-shelf OCR systems (e.g. Scansoft's OCR and several others [14]). However, as seen in Section 3.1 they can be effectively solved using a neural network that is trained on HIP characters. In this section we explore the abilities of such a neural network [6] for solving single character recognition problems under a variety of distortions and clutter. These distortions are similar to those commonly employed in HIPs and are described below. In this study, to better understand the NN's capabilities, the difficulty of the recognition problem is driven much higher than what would be deemed appropriate for use in a HIP.

3.2.1 Single Character Recognition Using Machine Learning

Character-based HIPs employ a set of character distortions to make them hard to OCR using computers. The basic character transformations include translation, rotation (clockwise or counterclockwise), and scaling. Rotation is usually less than 45 degrees to avoid converting a 6 into a 9, an M into a W etc. Examples of these distortions are presented in figures 11, 12, and 13.

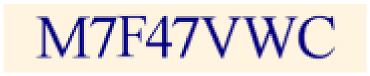


Fig. 11. Example of Plain Text (M7F47VWC)

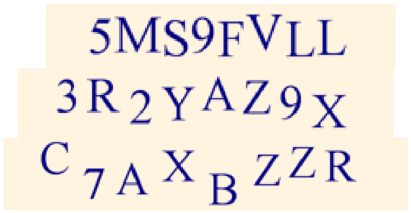


Fig. 12. Example of Translated Text, levels 10 (5MS9FVLL), 25 (3R2YAZ9X), and 40 (C7AXBZZR)

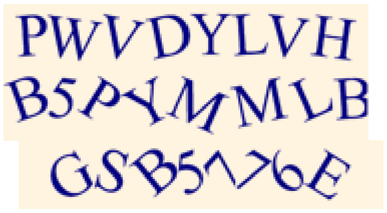


Fig. 13. Example of Rotation Text, levels 15 (PWVDYLVH), 30 (B5PYMMLB), and 45 (GSB5776E)

Both computers and humans find HIPs, using these three transformations, easy to solve. To increase the difficulty of computer-based OCR, we introduce two kinds of warp [17]:

1. **Global Warp**: The global warp produces character-level, elastic deformations (Figure 14). It is obtained by generating a random displacement field followed by a low pass filter with an exponential decay [17]. The resulting displacement field is then applied to the image with interpolation. These appear to bend and stretch the given characters. The magnitude of the warp is proportional to the total displacement distance of HIP image pixels. The purpose of these elastic deformations is to foil template matching algorithms.

2. **Local Warp**: Local warp is intended to produce small ripples, waves, and elastic deformations along the pixels of the character, i.e., at the scale of the thickness of the characters, rather than the scale of the width and height of the character (Figure 15). The local warp deformations are generated in the same manner as the global warp deformations, by changing the low pass filter cut-off to a higher frequency. The purpose of the local warp is to foil feature-based algorithms which may use character thickness or serif features to detect and recognize characters.

Crisscrossing straight lines and arcs, background textures, and meshes in foreground and background colors are common examples of clutter used in HIPs. In this paper, we used random arcs of different thicknesses as clutter.

Letter M under global warp

Left to right, letter M with varying amounts of global warp 75, 120, 180, 240, 300 respectively.

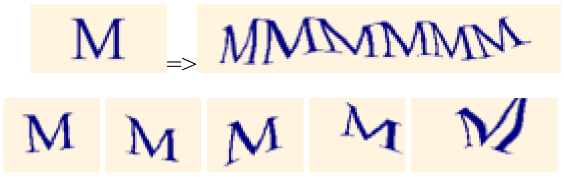


Fig. 14. Character transformation under global warp.

Letter M under local warp

Left to right, letter M with varying amounts of local warp 20, 40, 60, and 80, respectively.

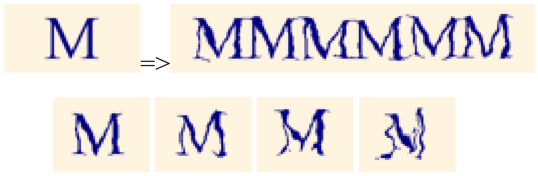


Fig. 15. Character transformation under local warp.

3.2.2 Single Character Recognition Using Machine Learning

We carried out a series of experiments to determine the recognition rates of the neural network classifier under the above distortions and clutter. In each experiment, a total of 110,000 random characters were sampled using the distortion and clutter settings. 90,000 characters were used for training and 10,000 were used for validation. Test error was computed over the remaining 10,000 characters. Thirty one characters from {A-Z, 0-9} were chosen. Five characters that can be easily confused were discarded. These were I, O, Q, 0, and 1. Characters were rendered in Times Roman font at font size 30.

In this paper, distortion and clutter were added to HIPs in the following order a) characters were rendered at random locations (with translation and rotation), b) foreground non-intersecting clutter (thick arcs that do not intersect) was rendered, c) foreground intersecting clutter (thin and thick foreground arcs that intersect), d) background intersecting clutter (thin and thick background arcs) and finally e) global and local warps were applied.

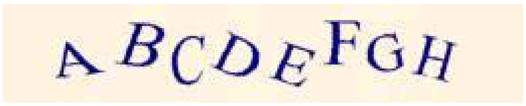


Fig. 16. Example of Baseline rotation, translation and scale for studying local warp and arc clutter.

The first three experiments studied rotation, global warp, and local warp in isolation. In the first experiment, characters were rotated randomly over the range of -45 to +45 degrees. In the second experiment, global warp was varied from 120 to 360 in steps of 60. For each setting a new neural network was trained. The third experiment studied recognition accuracy as local warp was varied from 20 to 80 in steps of 20. The fourth experiment was designed to explore the combined effect of simultaneously using these distortions and to provide a baseline for clutter experiments. The baseline setting used 80% - 120% scaling (x-, y-, or both), -20 to +20 degrees of rotation, a global warp setting of 75, and a local warp setting of 20. The fourth experiment used the baseline setting and the local warp was varied from 20 to 80 in steps of 20.

Single character neural network recognition results (on the test set) under different types of distortion are presented in Table 1. The NN accuracy was completely immune to rotation (with zero percent error), followed by local and global warps. The highest error rate was 8.08% for a Global warp of 360. A marginal increase (less than 2x) in the error rate is observed when multiple distortions are applied at moderate levels but the error rates still stay low (< 6%). With all error rates less than 10%, it is clear that the NN is very effective at recognizing characters even in the presence of significant distortions.

Table 1. Computer single character recognition error rates for affine transformations and warp. The baseline setting uses 80% - 120% scaling (x-, y-, or both), -20 to +20 degrees of rotation, and a global warp setting of 75.

Distortion (parameter range)	Computer Error Rates
Rotation (-45° to 45°)	0.00%
Global warp (120, 180, 240, 300, 360)	0.04%, 0.29%, 2.40%, 4.81%, 8.08%
Local warp (20, 40, 60, 80)	0.01%, 0.04%, 0.54%, 3.51%
Local warp (20, 40, 60, 80) + Baseline	0.01%, 0.22%, 1.19%, 5.23%

Experiments five through nine investigated single character computer recognition accuracies in the presence of baseline setting with varying degrees of arc clutter. Foreground arcs are rendered in the same color as characters and are designed to join adjacent HIP characters together. Background arcs are rendered in the background color and as such are designed to break up characters into disconnected pieces. Both foreground and background arcs are of constant thickness. Two levels of arc thickness were chosen with thin arcs being 2 pixels wide and thick arcs being 4-5 pixels wide. The combination of thin and thick arcs were chosen to model the thin and thick portions of characters in the Times font. The number of arcs, N_{arcs}, rendered on or around the character was determined by the arc density, D, using:

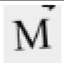
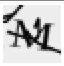
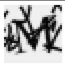
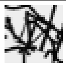
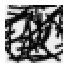
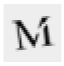
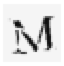
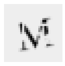
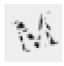
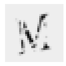
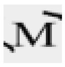
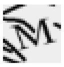
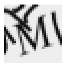
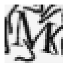
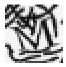
$$N_{arcs} = ceil\left[WH(D/S)^2\right] \quad \textbf{(1)}$$

where W and H are the width and height of the HIP image, respectively. S is the font size and *ceil* is the ceiling function. Example 8-character HIPs using these distortions and clutter are presented in Section 4.1. One character HIPs were generated with the same margin as in these figures with a 40 pixel x 40 pixel image centered on the character being used for recognition. Single character recognition experiments used a single character rendered on a 90x50 HIP image at font size 30.

Table 2. Computer single character recognition error rates in the presence of clutter. The baseline setting uses 80% - 120% scaling (x-, y-, or both), -20 to +20 degrees of rotation, global warp at 75, and local warp at 20.

Type of Clutter	Arc Density				
	0.5	1.0	1.5	2.0	2.5
Thin Foreground Arcs + Baseline	0.04%	0.19%	0.75%	1.62%	3.07%
Thick Foreground Arcs + Baseline	0.27%	2.08%	6.11%	22.95%	34.04%
Thin Background Arcs + Baseline	0.00%	0.00%	0.01%	0.00%	0.06%
Thick Background Arcs + Baseline	0.01%	0.10%	0.19%	0.47%	1.16%
Thick Non-intersecting Foreground Arcs + Baseline	0.16%	0.29%	0.36%	0.22%	0.30%

Table 3. Sample images with thick foreground and background arcs.

Type of Clutter	Arc Density				
	0.5	1.0	1.5	2.0	2.5
Thick Foreground Arcs + Baseline					
Thick Background Arcs + Baseline					
Thick Non-intersecting Foreground Arcs + Baseline					

Single character neural network recognition results on the test set are presented in Table 2. Interestingly, the neural network does best against thick background arcs and thick non-intersecting foreground arcs with error rates under 0.5%. The neural network also does well in the presence of thin arcs (foreground and background) with error rates staying below 3.1% even when the arc density is as high as 2.5. In the presence of thick arcs, the neural network does well for arc densities up to 1.5, but dramatically deteriorates at higher densities. Table 3 presents examples of input images containing letter 'M' with thick foreground arcs at different arc densities. Images with thick non-intersecting arcs are also presented for reference. When arc density exceeds 1.5 for thick intersecting foreground arcs, significant parts of the character are lost making them unreadable. This explains the dramatic drop in neural network accuracy. These results clearly indicate that the neural network is very effective at recognizing characters even in the presence of significant arc clutter.

4 Human-Friendly HIPs

Human-friendliness of a HIP encompasses both a) the visual appeal and annoyance factor of a HIP, and also b) how well it utilizes the difference in ability between humans and machines at solving segmentation and recognition tasks. As in the case of HIP security, human-friendliness is affected by each of the HIP design choices (Section 2.2) to different degrees. In general, the annoyance factor increases with increasing HIP length. Most online HIPs do not use more than 8 characters. A dictionary of words or a phonetic generator can make it easy for humans to type in the HIP solution as if it were a pseudo-word. Also, phonetic generators may help in reducing transcription and typing errors. Background and foreground textures and colors help blend the HIP image into the web page (or UI) theme and make it appear less intrusive or taxing to the user.

Many of the design choices that make HIPs human-friendly tend to reduce HIP security and vice versa. However, this is not always the case. Some attributes such as colors, textures, and anti-aliasing have negligible effect on security while they can significantly improve appeal. This might be the reason why MSN, Yahoo v2, and Google use two color HIPs (with varying gray levels). Further, human-friendliness can be affected by factors that have no impact on HIP security, such as the display size and color of the HIP image.

4.1 HIP User Studies

Human-friendliness is best studied through user studies with human participants. Three sets of user studies were conducted to understand human abilities in segmenting and recognizing characters in HIPs. The first set described in Section 4.1.1 explores human accuracy under rotation, scaling, local and global warp independently. The baseline combination of parameters with varying local warp levels was also added to the first set. The second and third sets explore human accuracy in the presence of foreground and background arc clutter and are described in Sections 4.1.2 and 4.1.3, respectively. The studies were designed to be run electronically, allowing participants to do the HIP recognition tasks from the comfort of their own offices. To improve efficiency of these user studies, 8-character HIPs were used. Human accuracy was defined as the percentage of characters correctly recognized. For example, for 8 character HIPs, getting on average 7 characters correct would imply an accuracy of 87.5 percent. The interested reader is referred to [18] for further details on these studies.

As in the case of the computer OCR experiments (Section 3.2.2), parameter settings were not limited to only those ranges that are useful for HIP design. Our goal was to understand human abilities across the board from settings where humans get almost 100% correct to those where they get less than 25% correct[4].

[4] For an 8-character HIP a 25% single character accuracy means that on average users can read about 2 out of the 8 characters in the HIP.

4.1.1 Human-Friendliness of Distortions

Seventy six users were recruited to participate in the first set of experiments. All were employees at a large software company. Average age of the participants was 35.2 (range of 22-54 years of age), 17 were female, and all but 12 had normal or corrected-to-normal vision. In addition, 42 wore glasses or contacts of some kind. All but six of the participants had at least an undergraduate education. The HIP parameter settings and ranges were the same as in the computer experiments for distortion. Only one HIP was created at each parameter level, and each participant saw that same exact HIP in a predetermined, randomized order. The seven parameters tested in the first user study were plain (or undistorted) text, translated test, rotated text, scaled text, global warping, local warping, and local warping combined with all the previous parameters at baseline settings (i.e., local warp + baseline).

A total of 68 8-character HIPs were presented to each subject. If the HIP was deemed to be unreadable by the participants, they could enter "unreadable" by pressing a button provided on the website for that trial. Each response provided by the participant was recorded. Total time to complete the experiment was approximately 15 minutes. Sample HIPs from these user studies are presented below. The numbers in parentheses indicate the level.

Plain, Translated, Rotated, and Scaled Text: Participants were very accurate at identifying the plain text letters (Figure 11). 73 participants recognized all letters perfectly, while 3 participants missed a single letter. We conjecture that these were transcription errors. The amount of translation was increased in nine steps from 0% to 40% of the character size (Figure 12). Participants had a very high accuracy rate with translated text. The accuracy rate was 99% or above for all levels. We rotated text in ten incremental steps from 0 degrees to 45 degrees (Figure 13). Participants had a very high accuracy rate with rotated text. The accuracy rate was 99% or above for all levels. Scaled text is text that is stretched or compressed in the x-direction as well as in the y-direction. Text was scaled in eleven incremental steps from 0% to ±50%. Participants had a very high accuracy rate with scaled text. The accuracy rate was 98% or above for all levels.

Global warp: Global warp covers the entire eight-character HIP. We increased the amount of global warping in 11 incremental steps from 0 to 390, as shown in Figure 18. Participants had a very high accuracy rate with levels of global warp up to level 270. Accuracy drops off dramatically with more global warp. A One-Way ANOVA shows that accuracy is reliably different for levels of global warp, $F(10,65) = 73.08$, $p < .001$. Post-hoc tests show that the 0-270 levels of global warp are reliably different from the 300-390 levels of global warp at the $p < .05$ level, using Bonferroni corrections for multiple tests in this and all following post-hocs.

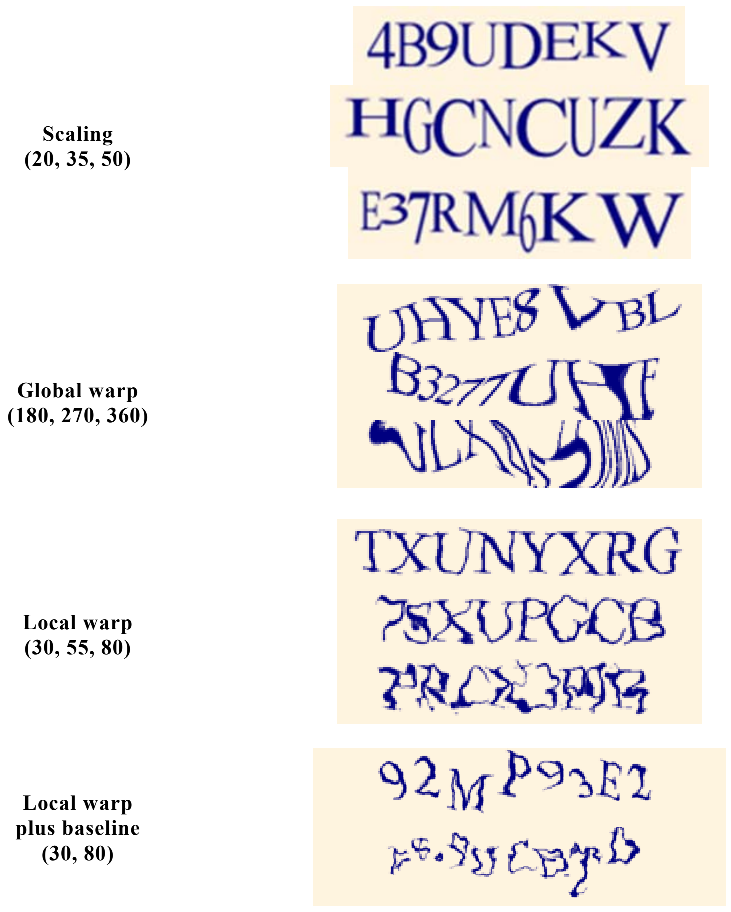


Fig. 17. Example of Baseline rotation, translation and scale for studying local warp and arc clutter.

Local warp: The local warp was incremented in 16 steps from 0 to 90, as shown in Figure 19. The local warp value indicates the magnitude of the local warp field and is proportional to the average movement of ink pixels in the HIP. Participants had a very high accuracy rate with levels of local warp up to level 45, and very poor accuracy at level 70 and above. A One-Way ANOVA shows that accuracy is reliably different for local warp levels, $F(15,60) = 120.24$, $p < .001$. Post-hoc tests indicate that levels 0-60 are reliably different from levels 65-90.

Local warp plus baseline: The baseline setting is used and local warp is gradually increased. Participants had a high accuracy rate with local warp plus baseline up to level 55 of local warp, as shown in Figure 20. After level 50, accuracy decreased in gradual steps, as is shown in the figure. A One-Way ANOVA shows that accuracy is reliably different for different levels of local warp plus baseline, $F(15,60) = 98.08$, $p < .001$. Post-hoc tests show that levels 0-55 are reliably different from levels 70-90.

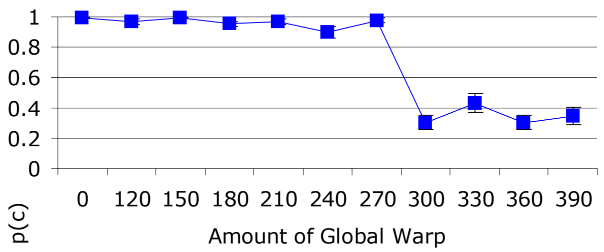


Fig. 18. Accuracy rate for global warp text

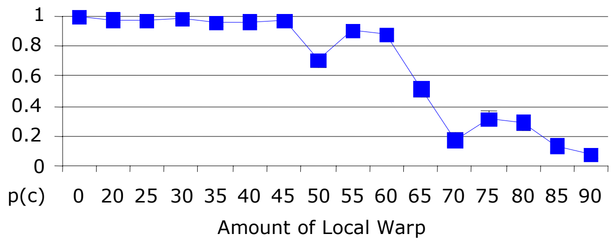


Fig. 19. Accuracy rate for local warp text

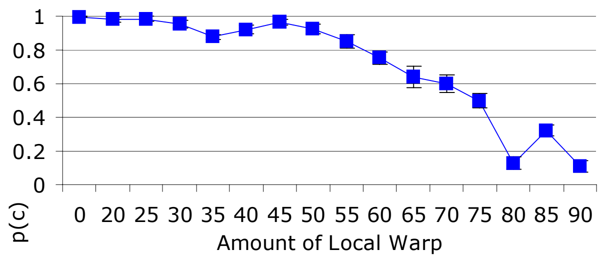


Fig. 20. Accuracy rate for local warp text with baseline

4.1.2 Human-Friendliness of Foreground Clutter

Twenty-nine more users from the same large software company were recruited for the second set of experiments. Average age of the participants was 35.2 (range of 26-54 years of age), 10 were female, and 23/29 had normal or corrected-to-normal vision. In addition, 19 wore glasses or contacts of some kind. All but six of the participants had at least an undergraduate education. Despite the similarities in the profiles between participants in studies 1 and 2, only one participant participated in both studies.

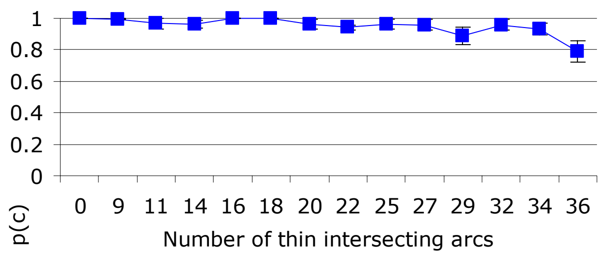


Fig. 21. Accuracy rate for thin intersecting arcs

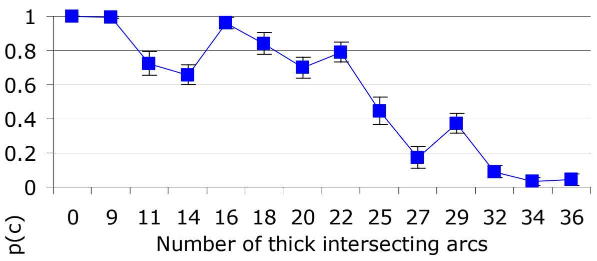


Fig. 22. Accuracy rate for thick intersecting arcs

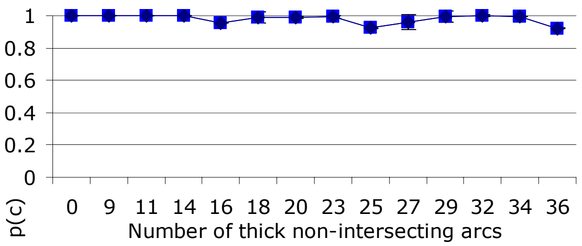


Fig. 23. Accuracy rate for thick non-intersecting arcs

The HIP parameter settings and ranges were similar to the ones in the computer experiments for clutter (Section 3.2.2). Only one HIP was created at each parameter level, and each participant saw that same exact HIP in a predetermined, randomized order. A total of 70 8-character HIPs with different types of arc clutter were presented to each subject. Other than the new HIP examples, all of details of the study were identical to Study 1. Sample HIPs from these user studies are presented in Figure 24.

Thin arcs that intersect plus baseline: There are 14 levels of arcs ranging from 0 to 36 arcs across the HIP, as shown in Figure 21. Participants had a high accuracy rate with thin arcs that intersect plus baseline, with accuracy above 90% for all but the highest number of arcs examined. A One-Way ANOVA shows that accuracy is reliably different for levels of thin arcs that intersect plus baseline, $F(13,16) = 2.70$, $p < .01$. Despite reliable main effects, post-hoc tests found no reliable differences between any two conditions.

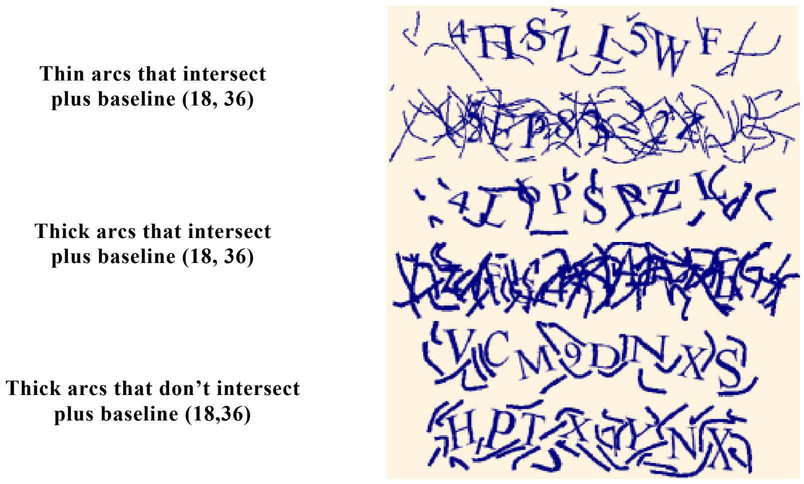

Thin arcs that intersect plus baseline (18, 36)
Thick arcs that intersect plus baseline (18, 36)
Thick arcs that don't intersect plus baseline (18,36)

Fig. 24. Sample HIPs from user study 1.

Thick arcs that intersect plus baseline: The baseline setting is combined with 14 levels of thick arcs that cross over the HIP characters, as shown in Figure 22. The number of arcs used ranged from 0 to 36. Not surprisingly, thick arcs that intersect are also difficult for participants when the baseline distortions are also incorporated. A One-Way ANOVA shows that accuracy is reliably different, F(13,16) = 49.27, p < .001. Post-hoc tests show that levels 0-22 are reliably different from levels 27-36.

Thick arcs that don't intersect plus baseline: The baseline is combined with 14 levels of thick arcs that do not cross over the HIP characters. The number of arcs used ranged from 0 to 36. Participants had a very high accuracy rates of 92% or above for all levels. The differences between levels of arcs was reliably different, F(13,16) = 2.12, p < .05. Despite the reliable main effect, post-hoc tests did not find any reliable differences between any two conditions (Figure 23).

4.1.3 Human-Friendliness of Background Clutter

Thirty-eight more participants from the same large software company were recruited for the third set of experiments. Average age of the participants was 33.4 (range of 22-57 years of age), nine were female, three were left-handed. All but five of the participants had at least an undergraduate education. Only one participant participated in one of the previous studies (study 2). The HIP parameter settings and ranges were similar to corresponding ones in the computer experiments (Table 3). A total of 12 8-character HIPs with different types of arc clutter were presented to each subject. Other than the new HIP examples, all of details of the study were identical to Studies 1 and 2. Sample HIPs from these studies are presented below

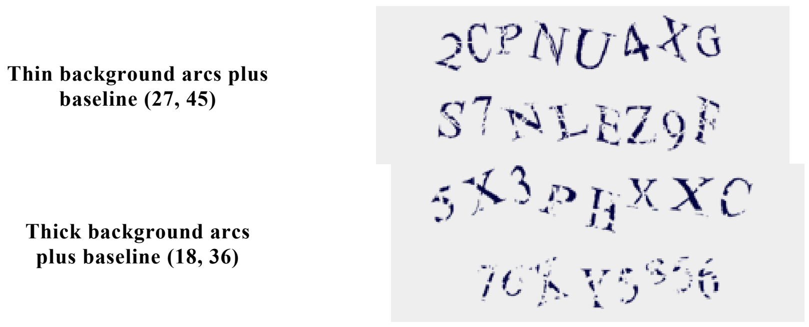

Thin background arcs plus baseline (27, 45)
Thick background arcs plus baseline (18, 36)

Fig. 25. Sample HIPs from user study 2.

Thin background arcs with baseline: In this condition, a HIP with the baseline settings is combined with 6 levels of thin background arcs that cross over the HIP characters. As shown in figure 26, background arcs in general break up characters into disjoint pieces. The number of arcs used ranged from 0 to 54. A higher number of background arcs were used as only background arcs that intersect characters show up on the HIP image. Participants had a very high accuracy rate with thin background arcs plus the baseline warp. The accuracy rate was 99% or above for all levels . The differences between the numbers of arcs is not reliably different, F(5,32) = 1.42, p > .05.

Thick background arcs with baseline: In this condition, a HIP with the baseline warp settings is combined with 6 levels of thick background arcs that cross over the HIP characters. Character breakup due to background arcs is more noticeable when thick arcs are used. The number of arcs used ranged from 0 to 54, as shown in Figure 27. Participants had a very high accuracy rate with thick background arcs except at the highest setting (54 thick arcs). The accuracy rate was 95% or above for levels up to 27 thick arcs and above 80% for higher levels up to 45. At level 54 the accuracy drops below 25% as characters become unreadable.

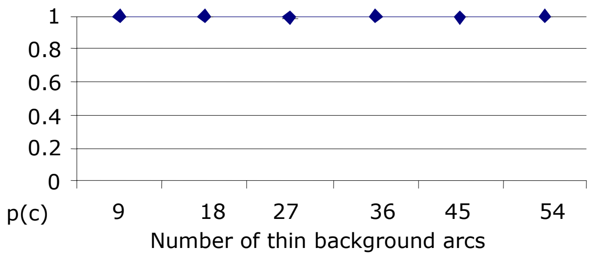


Fig. 26. Accuracy rate for thin background arcs

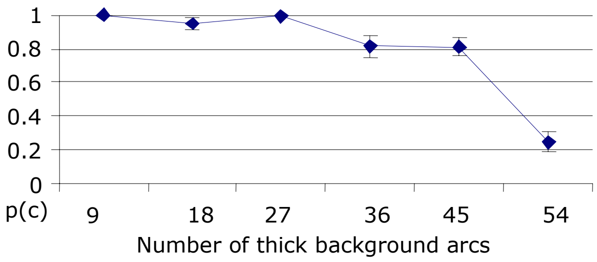


Fig. 27. Accuracy rate for thick background arcs

5 Building Better HIPs

5.1 Lessons Learned

The HIP breaking experiments have shown that many existing HIPs are pure recognition tasks and can be easily broken using machine learning. The stronger HIPs derive their strength from the segmentation part of the HIP challenge they pose, rather than the recognition problem. Recognition of characters in today's HIPs (given correct segmentation) is possible with high accuracy using convolutional neural networks [6].

Computer experiments on single character recognition have shown that neural networks effectively solve the recognition problem even when the distortion levels and clutter densities are driven very high. HIP user studies have shown that human recognition is good under translation, rotation, and scale variations with low to moderate levels of local and global warp. The human ability to solve HIPs is also good in the presence of thin foreground arcs, thick non-intersecting foreground arcs and thin and thick background arcs. However, humans do not do well in the presence of moderate to high levels of thick foreground arcs. Computers do relatively better in handling thick foreground arcs but also deteriorate significantly at high densities.

Comparing computer and human performance, we see that computers are better than humans at recognition (when segmentation is solved). One should note that the user studies (Section 4.1) required humans to solve both the segmentation and recognition problems whereas single character computer recognition experiments (Section 3.2) only required the neural network to solve the recognition problem[5]. In the distortion experiments and at low clutter densities these two results can be directly compared (as segmentation is trivial in these cases). However, at high distortion and clutter densities such a comparison would not be valid. We plan to study these scenarios more directly in future experiments and HIP user studies.

Segmentation is intrinsically difficult for both computers and humans because:

1) Segmentation is computationally expensive. In order to find valid patterns, a recognizer must attempt recognition at many different candidate locations.

[5] We do not expect human performance to improve significantly if characters were pre-segmented. This is due to humans being extremely good at segmentation.

2) The segmentation function is complex. To segment successfully, the system must learn to identify which patterns are valid among the set of all possible valid and non-valid patterns. This task is intrinsically more difficult than classification because the space of input is considerably larger. Unlike the space of valid patterns, the space of non-valid patterns is typically too vast to sample. This is a problem for many learning algorithms which yield too many false positives when presented non-valid patterns.
3) Identifying valid characters among a set of valid and invalid candidates is a combinatorial problem. For example, correctly identifying which 8 characters among 20 candidates (assuming 12 false positives), has a 1 in 125,970 (20 choose 8) chances of success by random guessing.

5.2 Segmentation Based HIPs

We can use what we have learned to build better HIPs. The HIP in Figure 28 was designed to make segmentation difficult and a similar version has been deployed by MSN Passport for hotmail registrations.

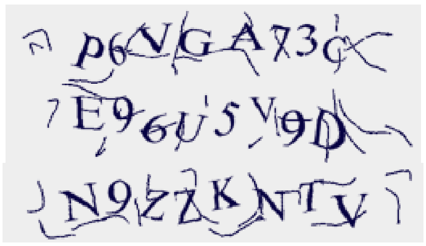


Fig. 28. Three samples of example segmentation HIP 1 (P6VGA73C, E96U5V9D, N9ZZKNTV).

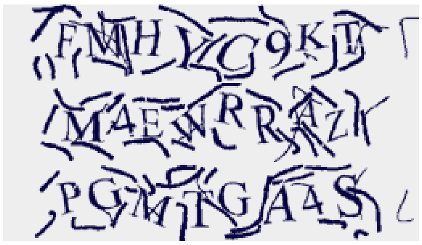


Fig. 29. Three samples of example segmentation HIP 2 (FMHYC9KT, M4EWRRAZ, PGMTGA4S).

The idea is that the additional arcs are themselves good candidates for false characters. The previous segmentation attacks would fail on this HIP. Furthermore, simple changes of fonts, distortions, or arc types would require extensive work for the attacker to adjust to. We believe HIPs that emphasize the segmentation problem, such as the above example, are much stronger than the HIPs we examined in this paper, which rely on recognition being difficult. Pushing this to the extreme, we can easily generate HIPs shown in Figure 29.

Despite the apparent difficulty of these HIPs, humans are surprisingly good at solving them, as suggested by Figure 23, indicating that humans are far better than computers at segmentation (Section 5.3.2 has the details). This approach of adding several competing false positives can in principle be used to strengthen a HIP by posing difficult segmentation problems for hackers.

5.2.1 Segmentation HIP Security

To build an automatic segmentor, we could use the following procedure. Label characters based on their correct position and train a recognizer. Apply the trained recognizer at all locations in the HIP image. Collect all candidate characters identified with high confidence by the recognizer. Compute the probability of each combination of candidates (going from left to right), and output the solution string with the highest probability. This is better illustrated with an example.

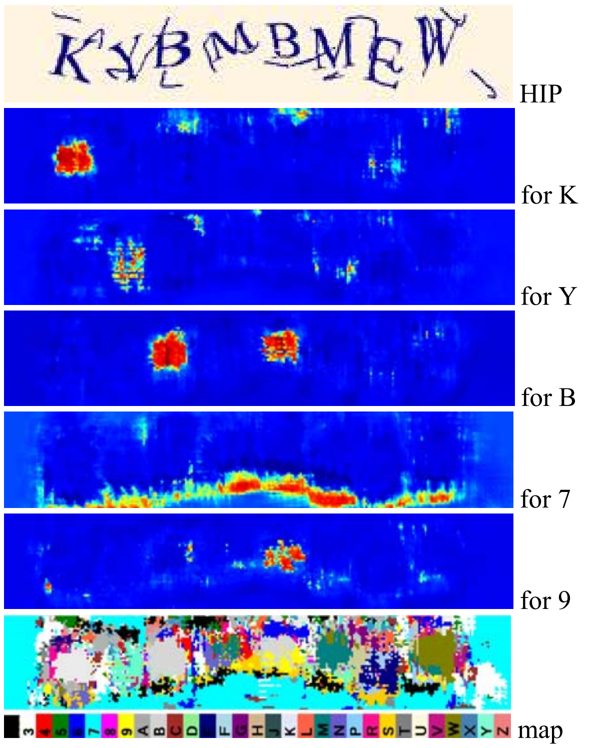


Fig. 30. Neural network output and combined map

Consider the HIP in Figure 30. After training a neural network with the above procedure, we have these maps (warm colors indicate recognition with high confidence) that show that K, Y, and so on are correctly identified. However, the maps for 7 and 9 show several false positives. In general, we would have a map for all the different candidates (see Figure 30).

We note that there are several false positives for each true positive. The number of false positives per true positive character was found to be between 1 and 4, resulting in a 1 in C(16,8) = 12,870 to 1 in C(32,8) = 10,518,300 random chance of guessing the correct segmentation for the HIP characters. These numbers can be improved upon by constraining solution strings to flow sequentially from left to right and by restricting overlap. For each combination, we compute a probability by multiplying the 8 probabilities of the classifier for each position. The combination with the highest probability is proposed by the classifier. We do not have results for such an automatic segmentor at this time. It is interesting to note that with such a method a classifier that is robust to false positives would do far better than one that is not.

5.2.2 Human-Friendliness of Segmentation Based HIPs

One user study was conducted to determine human-friendliness of the segmentation based HIPs presented in figures 28 and 29. The same set of 38 users from section 4.1.3 were used in these experiments. Ten eight-character HIPs of each type were used in the study. Other than the new HIP examples, all of details of the study were identical to the three user studies from Section 4.1. Participants had a very high accuracy rate for both segmentation based HIPs. Accuracy for HIP 1 was above 91% (Figure 28), while accuracy for segmentation HIP 2 was above 89% (Figure 29). Overall accuracy was slightly lower for segmentation HIP 2 examples, but the difference between the two was not statistically reliable, t(37) = 1.27, p = 0.21. The lack of any statistically significant difference indicates that both of these challenges are of somewhat equal difficulty to humans, though segmentation HIP 2 (Figure 29) poses a much harder segmentation problem for computers.

6 Conclusion

In this paper, we have successfully applied machine learning to investigate HIP security and we studied human reading ability under distortions and clutter common to HIPs. We have learned that decomposing the HIP problem into segmentation and recognition greatly simplifies analysis. Recognition on even unprocessed images (given segmentation is solved) can be done automatically using neural networks. Further, the HIP user studies have shown that given correct segmentation, computers are much better at HIP character recognition than humans. Segmentation, on the other hand, is the difficulty differentiator between weaker and stronger HIPs. Preliminary user studies on segmentation based HIP indicate that humans are just as good at solving segmentation based HIPs as they are at solving recognition based HIPs. In light of these results, we propose that segmentation based reading challenges are the future for building stronger human-friendly HIPs. The contribution of this research is to continue to drive HIP design from a user-centered perspective, wherein we try to design for a “sweet spot” that maximizes the comfort of human solvers while minimizing the ease of the code being broken through machine learning.

Acknowledgements
We would like to acknowledge Chau Luu for her help with developing the website for the user studies. We would also like to acknowledge Cem Paya, Erren Lester, Shannon Kallin, Julien Couvreur and Jonathan Wilkins in the MSN Passport team, for helping with the design, testing, and deployment of new segmentation based human-friendly HIPs. Finally we would like to thank Josh Benaloh from the MSR crypto group for not letting us compromise security.

References

1. Baird HS (1992), "Anatomy of a versatile page reader," *IEEE Proceedings*, v.80, pp. 1059-1065.
2. Chellapilla K, and Simard P, "Using Machine Learning to Break Visual Human Interaction Proofs (HIPs)," NIPS 2004, MIT Press.
3. *First Workshop on Human Interactive Proofs*, Palo Alto, CA, January 2002.
4. Von Ahn L, Blum M, and Langford J, *The Captcha Project.* http://www.captcha.net
5. Baird HS and Popat K (2002) "Human Interactive Proofs and Document Image Analysis," *Proc. IAPR 2002 Workshop on Document Analysis Systems*, Princeton, NJ.
6. Simard PY, Steinkraus D, and Platt J, (2003) "Best Practice for Convolutional Neural Networks Applied to Visual Document Analysis," in ICDAR'03, pp. 958-962, IEEE Computer Society, Los Alamitos.
7. Mori G and Malik J (2003), "Recognizing Objects in Adversarial Clutter: Breaking a Visual CAPTCHA," CVPR'03, IEEE Computer Society, vol.1, pages:I-134 - I-141, 2003.
8. Chew M and Baird HS (2003), "BaffleText: a Human Interactive Proof," *Proc., 10th IS&T/SPIE Document Recognition & Retrieval Conf.,* Santa Clara, CA, Jan. 22.
9. LeCun Y, Bottou L, Bengio Y, and Haffner P, "Gradient-based learning applied to document recognition,' *Proceedings of the IEEE*, Nov. 1998.
10. Selfridge OG. (1959). Pandemonium: A paradigm for learning. In *Symposium in the mechanization of thought process* (pp.513-526). London: HM Stationery Office.
11. Pelli DG, Burns CW, Farrell B, and Moore DC, "Identifying letters." (accepted) *Vision Research*.
12. Goodman J and Rounthwaite R, "Stopping Outgoing Spam," Proc. of the 5th ACM conf. on Electronic commerce, New York, NY. 2004.
13. Baird HS and Luk M, "Protecting Websites with Reading-Based CAPTCHAs," *Second International Web Document Analysis Workshop* (WDA'03); 2003 August 3; Edinburgh; Scotland.
14. Coates AL, Baird HS, and Fateman RJ, "Pessimal Print: A Reverse Turing Test," *Sixth International Conference on Document Analysis and Recognition (ICDAR '01)*, September 10 - 13, 2001, Seattle, WA.
15. Thayananthan A, Stenger B, Torr PHS, Cipolla R, "Shape Context and Chamfer Matching in Cluttered Scenes," CVPR (1) 2003: 127-133.
16. Moy G, Jones N, Harkless C, Potter R, "Distortion Estimation Techniques in Solving Visual CAPTCHAs," CVPR'04, Volume 2, pp. 23-28, June 27 - July 02, 2004, Washington, D.C., USA.
17. Deriche R, "Fast Algorithms for Low-Level Vision", *IEEE Trans. on PAMI*, 12(1), January 1990, pp. 78-87.
18. Chellapilla K, Larson K, Simard P, and Czerwinski M, "Designing Human Friendly Human Interaction Proofs (HIPs)," in Conference on Human factors In computing systems, CHI 2005. ACM Press.