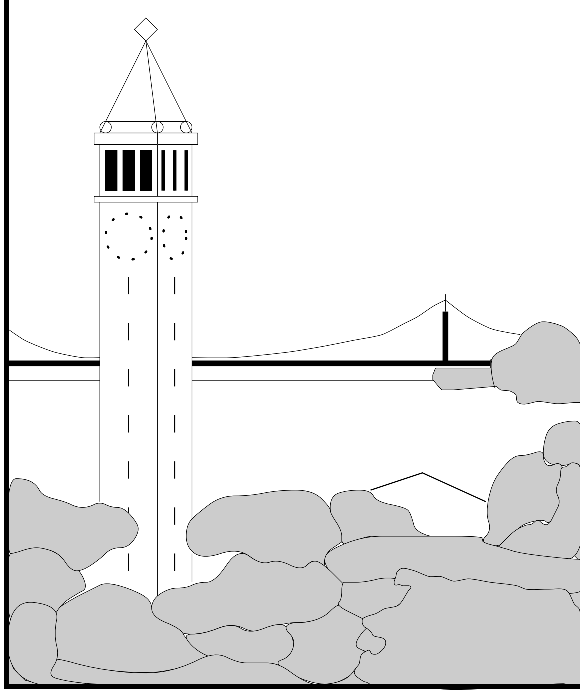
Image Recognition CAPTCHAs

Monica Chew
mmc@cs.berkeley.edu

J. D. Tygar[1]
tygar@cs.berkeley.edu



Report No. UCB/CSD-04-1333
USPS 102592-01-Z-0236

10 June 2004

Computer Science Division (EECS)
University of California
Berkeley, California 94720

Abstract. CAPTCHAs are tests that distinguish humans from software robots in an online environment [4, 18, 8]. We propose and implement three CAPTCHAs based on naming images, distinguishing images, and identifying an anomalous image out of a set. Novel contributions include proposals for two new CAPTCHAs, the first user study on image recognition CAPTCHAs, and a new metric for evaluating CAPTCHAs.

1 Introduction

We want to distinguish Internet communications originating from humans from those originating from software robots. Alan Turing's celebrated "Turing Test" paper [17] discussed the special case of a human tester who attempts to distinguish humans and artificial intelligence computer programs. However, the situation is far harder when the tester is a computer. Recent interest in this subject has spurred a number of proposals for CAPTCHAs: *Completely Automated Public Tests to tell Computers and Humans Apart* [18, 4]. This interest is motivated in large part by a variety of undesirable behavior associated with software robots, such as sending bulk unsolicited commercial e-mail (spam), or inflating ratings on a recommender system by rating the same product many times. Paypal and Yahoo already require CAPTCHAs in order to provide services. However, sufficiently advanced computer programs can break a number of CAPTCHAs that have been proposed to date.

Integrating a CAPTCHA into a software system raises a question which straddles the fields of *human computer interactions* (HCI) and *computer security*: how do we develop an effective CAPTCHA that humans are willing to take? This paper examines three proposals for image recognition CAPTCHAs and finds merits and weaknesses associated with each of them. Since CAPTCHAs straddle HCI and computer security, we need to apply techniques drawn from both fields to analyze them. This paper concentrates not only on the underlying security issues raised by these CAPTCHA proposals, but also on their usability. We validate our results with serious user tests.

CAPTCHAs must satisfy three basic properties. The tests must be

- Easy for humans to pass.
- Easy for a tester machine to generate and grade.
- Hard for a software robot to pass. The only automaton that should be able to pass a CAPTCHA is the one generating the CAPTCHA.

The first requirement implies that user studies are necessary to evaluate the effectiveness of CAPTCHAs. The second and third requirements push us in a different direction. We must find a test with a new property: the test must be easy to generate but intractable to pass without special knowledge available to humans and not computers. Image recognition seems to be such a problem. Humans can easily identify images, but to date, image recognition appears to be a hard problem for computers.

Generating tests is also a challenge. Unless the number of potential tests is huge or the material being tested is highly dynamic, one runs the risk of an adversary generating all possible tests and using a hash function to look up the answer in a precomputed database.

Note that CAPTCHAs need not be 100% effective at rejecting software robots. As long as the CAPTCHA raises the cost of using a software robot above the cost of using a human, the CAPTCHA can still be effective.

1.1 Contributions of this work

A number of researchers [4] have proposed the image recognition-based *naming CAPTCHA*, where the test subject is asked to identify a word associated with a set of images. We propose two variations on this approach:

- Asking the test subject to determine if two subsets of images are associated with the same word or not (the *distinguishing CAPTCHA*);
- Showing the test subject a set of images where all but one image is associated with a word and asking the test subject to identify the *anomalous* image (the *anomaly CAPTCHA*).

In this paper, we propose a new metric for evaluating CAPTCHAs, implement all three approaches, evaluate them both theoretically and in user studies, and find that anomaly identification appears to be the most promising approach. Preliminary results indicate that 100% of users can pass the anomaly CAPTCHA at least 90% of the time. We also help expand the literature on the intersection between HCI and computer security by discussing a number of unique HCI issues that arise in the evaluation of CAPTCHAs.

2 Survey of existing CAPTCHAs and attacks

Several types of CAPTCHA already exist. The types of tasks they require include: image recognition, text recognition, and speech recognition. All of the CAPTCHAs in use are either broken or insufficiently studied. The idea of using image recognition for CAPTCHAs is not new. However, no previous formal study of such CAPTCHAs exists. There are two prototypes of image recognition CMU's CAPTCHA website.[4] Both these prototypes use a small, fixed set of images and responses. Text-based CAPTCHAs require the user to transcribe an image of a word. For example, Yahoo relies on the EZ-Gimpy CAPTCHA, developed at CMU, to protect a against automatic registration of free email accounts[4]. EZ-Gimpy renders common English words in various FreeType fonts and degrades them (e.g., blurring, adding background grids or gradients) using the Gimp, an image-processing tool. Greg Mori and Jitendra Malik attack on EZ-Gimpy with a success rate of 83% [12], with full knowledge of font and dictionary.

2.1 Image-based CAPTCHAs

Pix presents the user with six images of the same subject and requires the user to identify the subject of the image. The number of query terms is small (less than twenty), and the image set for a particular query term is fixed. The Pix scheme is identical to the naming images presented here, but the Pix database of images and words is much smaller. Animal Pix is also a prototype, where the user must identify each of three animals, selecting the answers from a list of ten animals. Both of these prototypes were developed at CMU, and use a small fixed set of images and responses.

2.2 Text-based CAPTCHAs

Text-based CAPTCHAs require the user to transcribe an image of a word. All CAPTCHAs below are broken.

The EZ-GIMPY CAPTCHA Yahoo relies on a CAPTCHA called EZ-Gimpy, developed at CMU, to protect a against automatic registration of free email accounts[4] . EZ-GIMPY renders the words in various FreeType fonts and degrades them using the Gimp, an image processing tool[10]. EZ-Gimpy's image degradations include background grids and gradients, non-linear deformations, blurring, and additive pixel noise.

Greg Mori and Jitendra Malik attack on EZ-Gimpy with a success rate of 83% [12]. They use generalized shape contexts to find candidates similar to the target shape and prune the tree of candidate words using a lexicon.

The PessimalPrint CAPTCHA Another example of a reading-based CAPTCHA is PessimalPrint [5]. The words are between 5 and 8 characters long, with no ascenders (except for *i*) or descenders, to defend against character shape-coding OCR[16]. The motive for the length restriction is that short words are more subject to brute-force template matching attacks, and long words are more vulnerable to word-shape recognition (and more burdensome for humans). This approach yields a small, fixed lexicon. There are over a thousand words in the standard Unix dictionary that meet the length and ascender-descender restrictions, but many of them are uncommon: in the end, PessimalPrint used only 70 words.

PessimalPrint degrades word images according to the *Baird degradation model*, simulating physical defects caused by copying and scanning of printed text[1].

Mori and Malik used their EZ-Gimpy attack on PessimalPrint, with the same parameters as in the EZ-Gimpy attack and with a known font and dictionary. The attack correctly deciphered 40% of the images.

The Altavista CAPTCHA Altavista, which no longer provides email services, also relied on CAPTCHAs to prevent automated account registrations. The Altavista CAPTCHA, like the Paypal CAPTCHA, also uses a random string of letters and digits. The Altavista CAPTCHA rendered each character in a different font with a skewed baseline. Ari Juels can break this CAPTCHA with probability 1/500. [15]

3 Image Recognition CAPTCHAs

We use the following three tasks to construct our CAPTCHAs.

The naming CAPTCHA **The anomaly CAPTCHA**

Fig. 1. The figure on the left is the naming CAPTCHA. (Answer: *astronaut*.) The right figure illustrates the anomaly CAPTCHA. Five images are of *moose*, top middle anomalous image is of *weave*.

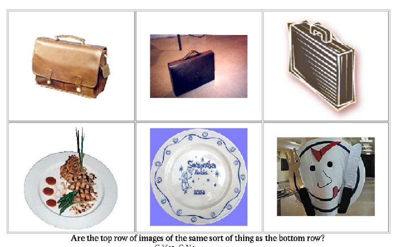


Fig. 2. The distinguishing images CAPTCHA. The top row of images is of *briefcase*, and the bottom row is of *plate*.

1. **The naming images CAPTCHA.**
   The *naming CAPTCHA* presents six images to the user. If the user correctly types the common term associated with the images, the user passes the round. Figure 1 shows an example of one round of the naming CAPTCHA. The common term illustrated in the figure is *astronaut*.
2. **The distinguishing images CAPTCHA.**
   The *distinguishing CAPTCHA* presents two sets of images to the user. Each set contains three images of the same subject. With equal probability, both sets either have the same subject or not. The user must determine whether or not the sets have the same subject in order to pass the round. For example, of six images, the subject of the top three images could be *briefcase*, and the subject of the bottom three images could be *plate*.
3. **The identifying anomalies CAPTCHA.**

The *anomaly CAPTCHA* presents six images to the user: five images are of the same subject, and one image (the anomalous image) shows a different subject. The user must identify the anomalous image to pass the test. Figure 1 shows five images of *moose* and one image of *weave*.

To make an image recognition CAPTCHA, we need a database of labelled images, several images for each label in a dictionary. Since it will be easy for humans to guess the label and hard for machines, we can use this database to create CAPTCHAs.

Dictionary For the labels, we use a dictionary of 627 English words from Pdictionary [14]. Pdictionary is an pictorial dictionary, so every word in it is easy to illustrate. Easily illustrated or visualized words are important; pictures of abstract nouns such as *love* or *agony* are difficult to identify.

Images For the images, we collect the first 20 hits from Google's image search on each word in the dictionary, using moderate safe search (a content filter). After culling broken links and potentially offensive images, 10116 images remain.[2] For this study, we pre-fetched the images to minimize latency when presenting the user with a round (in order to get more accurate timing information) and to ensure that no images are offensive. However, one could easily fetch the images dynamically to take advantage of changes in Google's image index.

3.1 Problems affecting human CAPTCHA performance

Each of the three above tasks (naming images, distinguishing images, identifying an anomalous image) incur potential problems that might lower test scores for humans. A problem lowers the probability that a human user will pass. A potential problem might hurt or help human performance. Below, × indicates a problem, ∘ a possible problem.

Task	Misspelling	Synonymy	Polysemy	Mislabelling
Naming images	×	×	∘	×
Distinguishing images			∘	×
Identifying anomalies			∘	×

Misspelling The human user may misspell a word.

- *The naming CAPTCHA*: The human user must type the word associated with the images. The user may misspell a word, failing the CAPTCHA even though the user knows the correct label of the image. *Solution*: We do not require an exact string match between the answer and the query in order for the user to pass. We choose Oliver's similarity score, a good metric that is also available as a PHP built-in function [13].
- *The distinguishing CAPTCHA and the anomaly CAPTCHA*: Misspelling is not a problem, since the user does not have to type anything.

Synonymy One word may have multiple correct definitions.

[2] Google has since added a strict content filter, which might help weed out more offensive images.

Fig. 3. Cup or Mug? The synonymy problem.

- *The naming CAPTCHA*: Synonymy will lower user scores. For example, a user may label a picture of a mug as a *cup* or a *mug* (Figure 3). Both answers are correct, but the *cup* answer will cause the user to fail. *Solution*: The CAPTCHA could use a thesaurus, and accept all synonyms of the expected answer. The problem with this solution is that an adversary also has a greater chance of passing if there is more than one correct answer.
  Another solution is to allow the user to fail some rounds: a CAPTCHA could be composed of m rounds, and a user would only have to pass k of them. We choose this solution and discuss picking the optimal k and m in Section 4.
- *The distinguishing CAPTCHA and the anomaly CAPTCHA*: Synonymy is not a problem since the user does not have to type anything.

Polysemy One word may correspond to multiple definitions. For example, a mouse is both a rodent and an electronic pointing device.

- *The naming CAPTCHA*: Polysemy might cause lower test scores, or it might raise them. On one hand, polysemy may provide more hints to the user. In the example above, if the user were only shown the picture of the rodent, the user might answer *rat* or *mouse*. However, if the picture of the pointing device is included, the user could rule out *rat* and give the correct answer. On the other hand, if the user does not know the alternate definition of mouse, the picture of the pointing device may be confusing.
- *The distinguishing CAPTCHA*: Polysemy could be a problem. For example, suppose the word is *mouse*. If the first set of images is of a rodent, and the second set is of a pointing device, the user will be confused. However, we expect this to be negligible in practice.
- *The anomaly CAPTCHA*: Polysemy could be a problem, if the user does not know the secondary definition of a word.

Solution: In each of the three CAPTCHAs, if polysemy is a problem, we can allow the user to fail some rounds.

Mislabelling Google indexes images by the name of the image. Some images on Google are labelled incorrectly. The label may have a meaning to the author of the web page, but no discernible connection to the image content to anyone else. For example, someone might have a pet cat whose name is Pumpkin, and thus an image search of the word pumpkin might produce pictures of a cat.

- *The naming CAPTCHA*: Mislabelling confuses the user.
- *The distinguishing CAPTCHA*: The distinguishing CAPTCHA presents two sets of images to the user, each set containing three images. Mislabelling will confuse the user, especially if the majority of images in a set are mislabelled.
- *The anomaly CAPTCHA*: A mislabelled image is indistinguishable from an anomalous image. Suppose the two labels are *pumpkin* and *briefcase*, and there is a pet cat named Pumpkin. There might be four easily identifiable pictures of a pumpkin and two anomalous images: one picture of Pumpkin the cat and a picture of a briefcase.

Solution: In each of the three CAPTCHAs, we can allow the user to fail some rounds. Additionally, in the naming CAPTCHA, we present six images of the same query to the user, with the hope the majority of the images are labelled correctly and the user will be able to guess the correct response.

Allowing the human user to fail some rounds alleviates all of these problems; we turn to the question of the total number of rounds the user must take, and the number of those the user must pass.

4 CAPTCHA metrics

In this section we choose two metrics for evaluating CAPTCHAs: a metric that allows us to measure CAPTCHA efficacy with respect to the number of rounds, and a metric measuring the expected time for a human user to take a CAPTCHA.

We can optimize a variety of CAPTCHA parameters:

1. **Type of task**. We can choose the nature of the tasks: naming the common element in a set of images, distinguishing images, and identifying an anomalous image.
2. **Number of images shown**. We can choose the number of images shown in each round. We chose to show six images, based on favorable response during the first round of experiments (Section 6).
3. **Number of rounds**. We can consider the number of rounds that compose a single CAPTCHA, and the minimum (threshold) number of rounds that a subject must pass to pass the CAPTCHA.
4. **Attack models**. We consider different attack models for a computer program attempting to pass the CAPTCHA. Section 5 discusses the effect of programs with good image identification ability which could do better than chance.

We consider several factors in choosing optimal values for the number of rounds and the threshold number of rounds that a subject must pass. First, human subjects have limits of how many rounds they are willing to tolerate. A human subject may find five rounds acceptable but is unlikely to agree to five hundred rounds. Second, computers have a speed advantage over humans. A computer can guess more quickly than a human can take a test. Below, we assume that within the time it takes for a human to complete one round, a computer can complete n rounds.[3]

CAPTCHA efficacy metric. We propose a new metric for CAPTCHA efficacy: the probability that in the time it takes a human to take a CAPTCHA, the human will pass and a computer will not. Let p be the probability that a human user will pass a round, q be the probability that a computer will pass a round, n be the number of times faster a computer is than a human, m be the number of rounds, and k be the threshold number of rounds. Then the efficacy metric G is

$$G = \sum_{i=k}^{m} \binom{m}{i} p^i (1-p)^{m-i} \cdot \left[1 - \sum_{i=k}^{m} \binom{m}{i} q^i (1-q)^{m-i} \right]^n$$

We also consider the expected time to complete a CAPTCHA. Short-circuit grading allows us to grade the CAPTCHA sometimes before all m rounds are complete. For a derivation of efficacy metric G and expected time to complete (pass or fail) a CAPTCHA with short-circuit grading, see the Appendix.

5 Attacking CAPTCHAs

Machine vision is a hard problem [2, 3, 7, 11]. A computer program is unlikely to correctly identify a previously unseen image without good training data. However, instead of the random-guessing attack model, a computer program could try the following attacks:

1. Build a database of labelled images from Google using our dictionary.
2. Search the database for the images presented in a round.

We can adopt the following countermeasures:

1. Use a large database of images.
   (a) Increase the size of the dictionary, so the image database is harder to maintain. Although the current dictionary (an online picture dictionary) was hand-picked by humans, adding more words to the dictionary does not necessarily require human intervention. For example, the most frequently-used search terms on Google's image search might be good candidates for inclusion in the dictionary.
   (b) Increase the number of images per word. For the user study, we prefetched the first 20 hits for each word in the dictionary. There is no reason not to increase this to, say, 100.
2. Use a dynamic database. Google updates its index of images frequently. There is no guarantee that an image in the database today will be there tomorrow. Also, recall that any attacker who indexes the entire Google image database is outside of our threat model, as we consider such attacks

[3] Some might argue that choosing any value n unnecessarily limits the computer program, but we could apply other countermeasures, such as temporarily disabling IP blocks if too many attempts fail in a particular period of time.

prohibitively expensive. Unlike the attacker, we do not have to maintain the database, since Google does it for us — the only advantage of prefetching images is speed.

3. Degrade the images, so searching for an exact match is no longer possible. This strategy might have the effect of lowering human performance. However, as long we maximize the gap between human performance and computer performance, we can still distinguish between the two. In addition, degrading the images is bound to have a negative effect on any kind of machine vision performance (both q and n). This is the first formal study of image recognition CAPTCHAs. We chose not to degrade the images in order to test the canonical *naming CAPTCHA*.

Recall that a CAPTCHA does not have to be 100% resistant to computer attack. If it is at least as expensive (due to database maintenance issues or image manipulations) for an attacker to break the CAPTCHA by machine than it would be to pay a human to take the CAPTCHA, the CAPTCHA is secure. Since these countermeasures are probably sufficient to deter current machine vision attacks, for the rest of the paper we only consider the random-guessing model.

6 First round of testing

Table 1. Similarity score examples. The similarity score is symmetric. All the words in the first column are from the dictionary used in the user study. All the words in the second column are actual answers from users.

Label	Answer	Score (%)
tomato	tomatoe	92.3
bone	bones	89.0
avocado	avacado	85.7
muffin tin	muffin	80.0
band-aid	bandage	71.4
ostrich	osterage	53.3
spade	cards	20.0

We performed two rounds of user testing. The first round had 4 users and was small and informal; the second round had 20 users who each completed 100 rounds of naming images and 100 rounds of identifying anomalies. This section is restricted to discussing the first, preliminary round of testing.

We had several goals in the first, informal round of user testing:

1. Parameterize CAPTCHAs and check for practical considerations.
   (a) Estimate p for the general population. If p for any task is too low, we reject that task.

(b) Using p, find the optimal m and k as in Section 4.
(c) Estimate the time to complete a round for the general population.
(d) Using p, m, and k, and round timing measurements, estimate the expected time to take a CAPTCHA. If the expected time for any CAPTCHA is too high, we reject that CAPTCHA.

2. Identify user interface problems.
3. Identify problems with the dictionary and database of images.

All tests were conducted on a web browser in the same location. Questions, answers, and timing measurements were logged on a `mysql` database on a single computer. During the testing, we received continuous feedback about user interface issues and difficulty of the tests.

Grading the naming CAPTCHA: For the naming images task, the Oliver similarity score is the best metric. We picked a minimum score of 80% to pass a round. For the preliminary test results, this lower bound allowed all the pluralization errors of the preliminary testers, and allowed no obviously wrong answers (no false positives). It did exclude a few obviously correct answers, but we wanted to be conservative to reduce the possibility of false positives. [4]

Problems with the dictionary and images: Mislabelled images were the most common problem. There are several mislabellings for surprising cultural reasons. For example, the word *cleaver* returns many pictures of cutting implements, and some pictures of the Cleaver family from the television program "Leave it to Beaver."

Optimal CAPTCHA parameters: We found the the optimal m and k for the experimentally determined values of p. For the naming CAPTCHA, the adversary is assumed to know the dictionary and guess randomly, so $q = 1/627$. For the distinguishing images CAPTCHA, the adversary picks uniformly at random whether or not the sets are same, so $q = 1/2$. For the anomaly CAPTCHA, the adversary picks the anomaly uniformly at random from a set of 6 images, so $q = 1/6$. Likewise, for the distinguishing CAPTCHA, $q = 1/2$.

We let $n = 100$ and searched exhaustively over values of m and k until $G \geq 95\%$ and m was minimized, as in Section 4. Although the optimal number of rounds (m) for the naming images varies between 4 and 6, the threshold number of rounds (k) that a user must take is 2. For the anomaly CAPTCHA, $m = 10$ and $k = 7$, an acceptable number of rounds. Unfortunately, for the distinguishing CAPTCHA, $m = 26$ and $k = 22$. Based on these rough timing measurements, completing one CAPTCHA should take no more than 2 minutes, even with the maximum number of rounds. (The formal testing shows that both CAPTCHAs take less than one minute.)

The distinguishing CAPTCHA is impractical: We no longer considered the distinguishing images CAPTCHA, since it requires 26 rounds to be effective.

[4] In the dictionary, no two words have a similarity score greater than 12%. Since the metric obeys the triangular inequality, there are no words that are 80% similar to two words in the dictionary. Therefore, the dictionary still has size 627 using this similarity score.

Test type	Median (%)	Mean (%)	Optimal m	Optimal k	t_{cor}	t_{inc}	E[t]
Anomaly ID	91	91.4 ± 12.3	7–15 (10)	6–9 (7)	6	13	51
Image ID	76.5	74.0 ± 19.2	3–8 (5)	2	8	11	24

Table 2. Average percent of 100 rounds passed (pass rate) by each user. The range in the optimal m column corresponds to the 95% confidence limits of the pass rate, and the number in parentheses indicates the optimal m for the mean pass rate. For example, the mean pass rate for anomaly detection rounds is 91%, with the 95% confidence interval 77.8-100%. The optimal m for the mean is 10, for the lower bound is 15, and for the upper bound is 7. Columns t_{cor} and t_{inc} correspond to the median number of seconds to pass or fail a round. E[t] is the expected number of seconds to take the CAPTCHA with short-circuit grading.

7 Second round of testing

20 users participated in the study ($N = 20$). The statistical significance of performance results varies with $\sqrt{N}$ and p. Based on results from the first round of testing, to achieve a 95% confidence interval of ±3% on p, we would have had to test nearly 1000 subjects, an infeasible number of subjects given financial and personnel constraints.

We recruited the users on an Internet bulletin board and with paper fliers. The users varied in age from 18 to 60, in education from 11th grade to PhD, and in frequency of computer usage from 0.5 to 40 hours a week. Four of the users are non-native speakers.

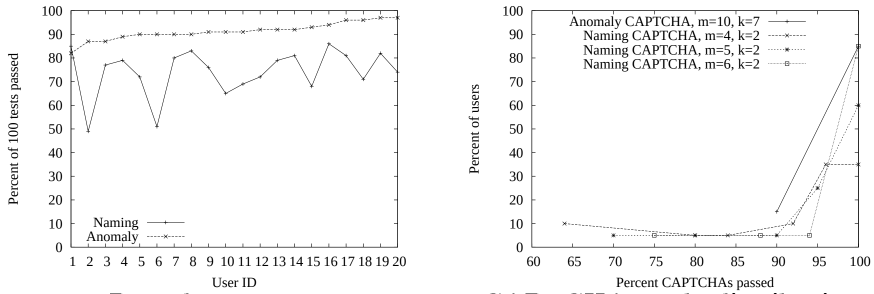
The performance results from a sample of 20 users may not predict of the performance of the entire population, in this case, the population of Internet users. The educational demographics of the sample is approximately the same as the demographics of the Internet from the most recent comprehensive survey, GVU's 10th WWW survey of 5022 users [9]. The GVU survey did not cover frequency of computer usage in hours per week, but we believe our sample is not skewed towards frequent computer usage. The median number of hours on a computer per week is 17.5, or 2.5 hours a day. Given that 20% of the participants used a computer on the job (40 hours/week) and this time includes email, web-surfing, shopping, word processing, 17.5 is a plausible median for the population.

The users were paid \$10-\$15 for completing 100 image identification rounds and 100 anomaly identification rounds, about the number of rounds that could be completed in an hour.The users rated the difficulty of each test, on a scale from 1 (easiest) to 5 (hardest). The rating process was not included in the timing measurements of each round. The users rated the test before the answer is revealed to prevent bias. The tests were conducted in the same location to eliminate network latency and variation in setup.[5] The users were also videotaped. After the tests, the users completed an exit interview.

We can learn several things from this graph of pass rates (Figure 4).

1. **The anomaly CAPTCHA is better than the naming CAPTCHA.** All but one person were better at identifying anomalies than naming images

[5] Two users completed the testing at their home, but their timing results are excluded.



Round pass rate **CAPTCHA grade distribution**

Fig. 4. The graph on the left gives the percent of 100 rounds passed (pass rate) by each user. The x axis corresponds to the user ID, and the y axis corresponds to the percent of rounds they passed for each type of test. For example, user 20 passed 97% of the anomaly detection rounds and 74% of the naming images rounds. The graph on the right gives the distribution of percent of CAPTCHAs passed. The x axis corresponds to percent of CAPTCHAs passed, and the y axis corresponds to the percent of users who passed that many CAPTCHAs.

because of mislabelled images and an imperfect grading scheme. Figure 5 illustrates that a lower similarity score cutoff would not help much. Either the subject knew the correct label but might have made one or two mistakes in spelling or pluralization (resulting in a sufficiently high similarity score), or the subject guessed the wrong label, occasionally because of synonymy. Improving the grading scheme requires solving the synonymy problem.

2. **CAPTCHA performance is not uniform across CAPTCHA types.** Subjects who were good at identifying anomalies are not necessarily good at naming images, and vice versa. This means that if user does poorly at one kind of CAPTCHA, we can switch to the other kind in hopes they do better. This switch should not benefit adversarial computer programs, since the parameters for both CAPTCHAs result in similar computer failure rates.
3. **No discrimination against a particular educational or technical level.** Occupation, computer skills, technical background and education do not seem to be very good indicators of performance based on our sample of $N = 20$ users. For example, of the two highest scorers (97%) in anomaly detection, one completed high school, and the other had a PhD. Although our sample is small, these results suggest that the two CAPTCHAs do not discriminate against a particular educational or technical skill level.
4. **The anomaly CAPTCHA is language-independent**. Speaking English natively seems to be a pretty good indicator of performance in the naming CAPTCHA, but not in the anomaly CAPTCHA.

Table 2 gives the median and mean percent of 100 rounds passed by each user (pass rate) for image and anomaly identification rounds. The upper and lower bounds denote the 95% confidence limits for $N = 20$. Table 2 also shows the upper and lower bounds for the revised m and k, given the 95% confidence limits.

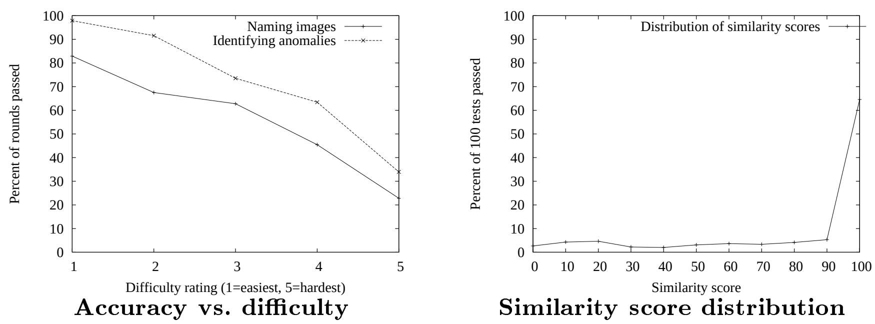


Fig. 5. The figure on the left gives accuracy vs. difficulty rating. The x axis corresponds to the difficulty rating, with 1 being the easiest rating and 5 the most difficult. The y axis is the mean pass rate for rounds with that rating. The figure on the right gives distribution of similarity scores. The x axis corresponds to the similarity score. The y axis is the proportion of naming images rounds in that bucket of scores. Each bucket is 10% wide, except the 100% bucket, which is 1% wide. We recommend a minimum similarity score of 80% to pass a round.

Note that the new values for m and k are identical to the old ones calculated after the preliminary testing.

Figure 4 shows the results of grading the rounds according to the parameters m and k. Using short-circuit grading produces nearly identical results (Section 4). From Figure 4, we see that 85% of the users passed the anomaly detection CAPTCHA 100% of the time with $m = 10, k = 7$, and 100% of the users passed the CAPTCHA at least 90% of the time. 85% of the users passed the anomaly detection 100% of the time with $m = 6, k = 2$. Table 2 shows the expected number of rounds and expected time to take a CAPTCHA. Figure 5 shows accuracy versus difficulty rating.

7.1 Lessons learned

At the end of the testing, each subject completed an exit interview. Mislabelled images were the most common complaint. The anomaly CAPTCHA is the best choice for the following reasons:

1. 90% of the users prefer it to the naming CAPTCHA.
2. 100% of the users pass the CAPTCHA at least 90% of the time, with $m = 10, k = 7$.
3. The expected time to take this CAPTCHA is 51 seconds.
4. Most people are willing to spend at least 2 minutes on a CAPTCHA.
5. The anomaly CAPTCHA is language-independent. Non-native speakers did equally well as native English speakers.

8 Open problems

CAPTCHAs are still a new research area. Open problems include

- The mislabelling problem. Of all the problems discussed in Section 3.1, mislabelling causes the most human errors. We may be able to solve this using collaborative filtering, where known human users rate images according to how well they evoke their label.
- Optimizing CAPTCHA parameters. We can present more images per round in the anomaly detection CAPTCHA to deflate computer performance.
- Testing other image recognition CAPTCHAs. For example, we could require the human user to locate Waldo in an image filled with background clutter and other Waldo-shaped objects.
- Improving usability. Because of the CAPTCHA "annoyance factor," not everyone in the study was willing to take a CAPTCHA, even in exchange for free services. Rounds rated difficult (primarily because of the mislabelling problem) are both more onerous and difficult to pass.

A CAPTCHA metrics

Table 3. CAPTCHA parameters.

p	Pr[human user passes one round]	m	Number of rounds in a CAPTCHA
q	Pr[computer passes one round]	k	Number of rounds needed to pass
n	Computer speed advantage		

CAPTCHA efficacy metric We propose a new metric for CAPTCHA efficacy: the probably that in the time it takes a human to take a CAPTCHA, the human will pass and a computer will not. Table 3 summarizes the parameters used in this section. Let p be the probability that a human user will pass a round.

Let m be the number of rounds in a CAPTCHA, and k be the number of rounds we require a user to pass. The probability that a human will pass the CAPTCHA is

$$H = \sum_{i=k}^{m} \binom{m}{i} p^i (1-p)^{m-i}$$

We identify a metric that allows us to compare different values for m and k, in particular one that measures the difference between human and computer success. Suppose that a computer is n times faster than a human, so it can try n rounds in the same time (Δt) that it takes a human to complete one. (We can simulate this by blocking an IP address if n requests come from it in the time it normally takes a human to complete 1 round.) Let q be the probability that a computer will pass a round. The the probability that a computer will not pass a CAPTCHA in Δt:

$$\bar{C} = \left[1 - \sum_{i=k}^{m} \binom{m}{i} q^i (1-q)^{m-i}\right]^n$$

We can combine H and $\bar{C}$ to create an efficacy metric G, the probability that given m, k, p, and q, in Δt, a human will pass and a computer will not:

$$G = H\bar{C}$$

G is monotonically increasing in m. Since we want m to be reasonably small for practical considerations, we can pick m and k such that $G \geq 95\%$ subject to the constraints that

1. m is minimized,
2. G is maximized.

The solution is unique — for a fixed p, q, and m, G first increases with k (since ΔG is dominated by monotonically increasing function $\bar{C}$), then finally decreases (since ΔG is dominated by monotonically increasing function H as a greater fraction of correct rounds is required). We call m and k that satisfy these constraints *optimal*.

Since p, q, and n can be determined experimentally (or estimated), all that remains is to choose the optimal m and k using metric G.

A.1 Minimizing user effort

Besides the practical consideration that m should be small, we want the expected time to take a CAPTCHA to be small. The expected time to complete a CAPTCHA is a function of the time to pass a round, t_p, the time to fail a round, t_f, and the probability of passing one round, p. We distinguish t_p from t_f because we hypothesize that either the human user will either know the answer immediately or be stumped. If the human user is stumped, the task is probably difficult, and the answer will probably be wrong. Thus, we expect a difference in time to complete a round with respect to accuracy. (The user study confirms this conjecture.)

The expected time to take all m rounds is

$$m \cdot [pt_p + (1-p)t_f]$$

Short-circuit grading We can minimize the expected time to take a CAPTCHA. In certain circumstances, we can determine the grade for a CAPTCHA before the user completes m rounds. For example, if p is nearly 1, and $m >> k$, it is very likely that a user will pass k rounds shortly after the kth round. In this case it is practical to allow the user to stop, up to $m - k$ rounds early. First, the user will be happier because the CAPTCHA takes less time and effort. Second, if an adversary is eavesdropping on the CAPTCHA testing (for example, to build a database of images), fewer images will be revealed. Similarly, the user is destined to fail the CAPTCHA if the user cannot possibly pass k rounds, given the number of rounds remaining and the number of rounds the user has currently passed.

Suppose we allow a user to stop taking a CAPTCHA either

1. once the user has passed k rounds (passing the CAPTCHA), even if the user has not completed all m rounds, or
2. once the user has failed $m - k + 1$ rounds (failing the CAPTCHA), even if the user has not completed all m rounds.

Using this short-circuit grading scheme does not negatively affect human user CAPTCHA performance. The scheme reduces the number of rounds taken in a CAPTCHA if and only if the grade has already been determined. Allowing or compelling the user to complete all of the rounds will not affect the grade for that CAPTCHA.

Let i be the first round such that the user has passed k rounds, thus passing the CAPTCHA. The user must pass round i (otherwise, the user would have passed the CAPTCHA at some round $j < i$). Thus, the user must have passed exactly $k - 1$ rounds by round $i - 1$. There are $\binom{i-1}{k-1}$ possibilities for this case, each possibility occurring with probability $p^{k-1}(1-p)^{i-1-k}$. We can now do a standard binomial distribution analysis.

The probability $p_{i,pass}$ that a user completes i rounds and passes exactly k of them is

$$p_{i,pass} = \binom{i-1}{k-1} p^k (1-p)^{i-k}$$

Similarly, let i be the first round such that the user has failed $m - k + 1$ rounds, thus failing the CAPTCHA. The probability $p_{i,fail}$ that a user completes i rounds and fails is

$$p_{i,fail} = \binom{i-1}{m-k} (1-p)^{m-k+1} p^{i-m+k-1}$$

We want to compute the expected time to take a CAPTCHA, where the user can stop as soon as the grade is determined. Let t_p be the median time for a user to pass a round, and t_f be the median time for a user to fail a round.

The time for the user to pass exactly k out of i rounds is

$$t_{i,pass} = k \cdot t_p + (i - k) \cdot t_f$$

and the time to fail exactly $m - k + 1$ out of i rounds is

$$t_{i,fail} = (m - k + 1) \cdot t_f + (i - m + k - 1) \cdot t_p$$

Thus, the expected time to take a CAPTCHA is

$$\sum_{i=k}^{m} (p_{i,pass} t_{i,pass} + p_{i,fail} t_{i,fail})$$

A.2 Estimating p for the whole population

The purpose of CAPTCHAs is to distinguish humans from computers in an online environment, so the ultimate user base of CAPTCHAs is the population of all Internet users. We want to find the task pass rate p for this population. However, it is impractical to recruit the entire population for our user study. We must estimate p using a population sample, the participants in the user study. How good our estimate of p depends on the size of this sample.

Let N be the number of subjects in the user study. Since the tests form a Bernoulli distribution, the standard deviation $\sigma = \sqrt{p(1-p)}$[6]. Then we can calculate the length of a 95% confidence interval centered about p as

$$L = z \cdot \frac{\sigma}{\sqrt{N}}$$

where $z = 1.96$ is the percentile of the normal distribution corresponding to a two-sided 95% confidence interval[6]. Thus we can be confident that 95% of Internet users have a task pass rate of $p \pm L$.

References

1. Henry S. Baird. Document image defect models. In H. S. Baird, H. Bunke, and K. Yamamoto, editors, *Structured Document Image Analysis*, pages 546–556. Springer–Verlag: New York, 1992.
2. K. Barnard, P. Duygulu, D. Forsyth, N. de Freitas, D. Blei, and M. Jordan. Matching words and pictures. *Special Issue on Text and Images, Journal of Machine Learning Research*, 3:1107–1135, 2002.
3. Kobus Barnard and David Forsyth. Learning the semantics of words and pictures. In *International Conference on Computer Vision*, volume 2, pages 408–415, 2001.
4. Manuel Blum, Luis A. von Ahn, John Langford, and Nick Hopper. The CAPTCHA Project. http://www.captcha.net, November 2000.
5. Allison L. Coates, Henry S. Baird, and Richard Fateman. Pessimal print: a reverse Turing test. In *Proc., IAPR 6th Intl. Conf. on Document Analysis and Recognition*, pages 1154–1158, Seattle, WA, September 2001.
6. Jay L. Devore. *Probability and Statistics for Engineering and the Sciences*. Brooks/Cole Publishing Company, 2nd edition, 1987.
7. A. Goodrum. Image information retrieval: An overview of current research. *Informing Science*, 3(2):63–66, February 2000.
8. Nicholas J. Hopper and Manuel Blum. Secure human identification protocols. In *Asiacrypt*, pages 52–66, 2001.
9. Colleen Kehoe, Jim Pitkow, Kate Sutton, Gaurav Aggarwal, and Juan D. Rogers. Gvu's 10th world wide web user survey. http://www.gvu.gatech.edu/user_surveys/survey-1998-10/, 1999.

10. Peter Mattis and Spencer Kimball. The GNU image manipulation program. http://gimp.org, 2002.
11. Sharon McDonald and John Tait. Search strategies in content-based image retrieval. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 80–87. ACM Press, 2003.
12. Greg Mori and Jitendra Malik. Recognizing objects in adversarial clutter: Breaking a visual CAPTCHA. In *Computer Vision and Pattern Recognition*, 2003.
13. J. J. Oliver. Decision graphs - an extension of decision trees. In *Proceedings of the Fourth International Workshop on Artificial Intelligence and Statistics*, pages 343–350, 1993.
14. Pdictionary. The internet picture dictionary. http://www.pdictionary.com, 2004.
15. Associated Press. Researchers battle web-bots with identity checks. www.siliconvalley.com/mld/siliconvalley/4797581.htm, December 2002.
16. A. L. Spitz. Moby dick meets geocr: Lexical considerations in word recognition. In *4th International Conference on Document Analysis and Recognition*, pages 221–232, Ulm, Germany, August 1997.
17. Alan Turing. Computing machinery and intelligence. In *Mind*, pages 433–60, 1950.
18. L. von Ahn, M. Blum, N. Hopper, and J. Langford. Captcha: Using hard AI problems for security. In *Eurocrypt*, 2003.