In Human Interactive Proofs: Second International Workshop (HIP 2005),
eds. H. Baird and D. Lopresti, Springer, May 2005, pp. 66-81

Collaborative Filtering CAPTCHAs

Monica Chew and J.D. Tygar*

University of California, Berkeley
{mmc,tygar}@cs.berkeley.edu

Abstract. Current CAPTCHAs require users to solve objective questions such as text recognition or image recognition. We propose a class of CAPTCHAs based on collaborative filtering. Collaborative filtering CAPTCHAs allow us to ask questions that have no absolute answer; instead, the CAPTCHAs are graded by comparison to other people's answers. We analyze the security requirements of collaborative filtering CAPTCHAs and find that although they are not ready to use now, collaborative filtering CAPTCHAs are worthy of further investigation.

1 Introduction

This paper proposes a framework for CAPTCHAs using collaborative filtering. By observing real-world trends made by human subjects, collaborative filtering CAPTCHAs attempt to extract complex patterns that reflect human choices. For example, humans who like a particular joke, such as a subtle pun, may also enjoy other jokes that incorporate similar patterns of whimsy, word-play, and ironic observation. We consider the proposition that these patterns are sufficiently complex that no computer agent can predict these patterns with equal accuracy. While one might naively believe that detecting patterns of humor is beyond the capability of any machine, we show in this paper that computer agents can do better than one might at first think. We conduct an experiment that demonstrates that joke-affinity CAPTCHAs can be weakly effective. Our results show that collaborative filtering CAPTCHAs, while not ready to use now, show promise beyond traditional CAPTCHA approaches and deserve further examination.

Why should we study collaborative filtering CAPTCHAs? Because current CAPTCHA research resembles an arms race between CAPTCHA developers and CAPTCHA attackers. CAPTCHA developers propose schemes which they hope are unbreakable, and CAPTCHA attackers break them. The text-recognition CAPTCHA EZ-Gimpy exemplifies this cycle [2]. EZ-Gimpy requires humans to transcribe an image containing a single English word. In 2003, Mori and Malik broke EZ-Gimpy with 87% success [10]. The Mori-Malik attack requires a dictionary. EZ-Gimpy designers then proposed a variation on EZ-Gimpy called

* This work was supported in part by the US Postal Service and the National Science Foundation. The opinions here are those of the authors and do not necessarily reflect the opinions of the funding sponsors.

H.S. Baird and D.P. Lopresti (Eds.): HIP 2005, LNCS 3517, pp. 66–81, 2005.

Gimpy-R which uses random character strings instead of English text, thus defeating a dictionary attack. Moy et al. broke Gimpy-R in 2004 with 78% success, and EZ-Gimpy using the same technique with 99% success [11].

Using collaborative filtering can make CAPTCHAs more difficult to break. Suppose we can develop a CAPTCHA where the attacker must derive CAPTCHA answers from other humans instead of solving an objective machine vision question such as text recognition. In theory, the only way to break such a CAPTCHA would be for the attacker to perform a user study and analyze the results, a very expensive proposition.

This paper contains the following contributions:

- We propose a collaborative filtering framework for CAPTCHAs.
- We propose new attack models on collaborative filtering CAPTCHAs.
- We give security requirements for input data to collaborative filtering CAPTCHAs.
- We present the results of an experiment on collaborative filtering CAPTCHAs.
- We give a list of open problems for further examination.

Section 2 describes collaborative filtering and its challenges. Section 3 outlines how to use collaborative filtering to build a CAPTCHA. Section 4 explains using Singular Vector Decomposition to predict user ratings. Section 5 describes experiments using the Jester dataset. Section 6 analyzes attacks on collaborative filtering CAPTCHAs. Section 7 lists related work, and Section 8 concludes with open problems.

2 Collaborative Filtering

Collaborative filters, or recommender systems, use a database of user preferences to predict items or topics a new user might like or find useful [3, 14]. For example, Amazon allows users to rate items for sale on a scale from 1 to 10. A new user, Alice, is compared to existing users based on purchase or browsing history. Amazon compares the user preferences of neighbors, or users who are historically similar to Alice, to predict new items Alice might like and then recommends them.

Challenges in collaborative filtering include:

- Accuracy. The prediction of Alice's ratings must be accurate in order for the recommendations to be useful. Additionally, in a collaborative filtering CAPTCHA, inaccurate predictions will cause humans to fail the CAPTCHA.
- Sparsity. For very large recommender systems (e.g., Amazon, the Internet Movie Database, and Ebay), even a very prolific user might have preference data for a very small percentage of the items in the system. Sparsity makes predictions more difficult.
- Scalability. Because nearest-neighbor algorithms (to find users with similar preferences) scale in the number of users and items, very large recommender systems suffer scalability problems.

– "Polluted" data. Malicious or apathetic users may enter incorrect preference data, hampering the accuracy of prediction algorithms. Dellarocas has done some work in this area to reject "outliers" — however, this approach has the unwanted effect of unfairly rejecting users with eccentric tastes [5]. In general, the collaborative filtering research concentrates on improving the predictive accuracy in the absence of adversaries, so this challenge is not well-studied in comparison with the rest.

In this work, we are primarily concerned with accuracy and polluted data.

3 Collaborative Filtering CAPTCHAs

Previous CAPTCHAs require users to solve cognitive tasks such as text recognition and image recognition. Both of these tasks are currently subjects of machine vision research. Sophisticated machine vision attacks exist for text-based CAPTCHAs; it is only reasonable to expect the machine vision community to make progress in image recognition if these CAPTCHAs are adopted.

What if we could use challenge questions that have no absolute answer? Then we could build a CAPTCHA where the user is correct so long as enough known humans agree. Collaborative filtering allows us to do so. Collaborative filtering is a way to aggregate data from many different human users so that we can easily compare new data. The collaborative filtering approach differs from previous approaches in that the CAPTCHA designer does not know the correct answer initially, but measures the correct answer from human opinions. There are many subjective topics we could use to build a CAPTCHA: however, finding a good source of input data is an open problem for reasons we discuss below.

3.1 Sources of Data

To build a collaborative filtering CAPTCHA, we require a source of data that evokes some aspect of our humanity that is difficult to quantify. For example:

– Humans recognize quality in art (such as movies, music, literature, or images), and computers do not.
– Art (visual art, music) evokes human emotion which may be unpredictable by computers.
– Humans have philosophical leanings (political opinions, religious doctrines, etc.) which are difficult to codify.
– Humans recognize humor in jokes, and computers do not.

Choosing a good source of data is difficult. For example, building a CAPTCHA out of movie ratings presents two problems: movies are time-consuming and expensive for users to watch and rate, and online oracles such as the Internet Movie Database can be used by adversaries. Cultural bias can also plague collaborative filtering CAPTCHAs: the filter may only make accurate predictions for certain demographic groups. However, all existing CAPTCHAs

(including those based on text and images) discriminate against a some demographic. Visually impaired or illiterate people cannot pass a reading-based CAPTCHA, for example.

3.2 Stages of the CAPTCHA

Given an appropriate topic that computers find difficult to evaluate, we can construct a CAPTCHA based on collaborative filtering. A collaborative filtering CAPTCHA goes through several phases:

- *Training.* In the training phase, a group of known humans rates documents in the gauge set. The seed data or training data will be used to generate predictions for new user who wish to take the CAPTCHA.
- *Testing.* The testing phase is entered whenever a new user wants to take a CAPTCHA. Suppose Alice wants to take the CAPTCHA. We present a strict subset of documents in the gauge set for Alice to rate. Based on Alice's ratings of the subset and the training data, we make predictions for Alice's ratings of the remainder of the gauge set. Alice then rates the remainder of the gauge set. We now have actual and predicted ratings for the remainder of the set. If the predicted ratings are close enough to the actual ratings, Alice passes. The threshold for passing is an open research problem.
- *Reseeding.* If Alice passes as human, the CAPTCHA enters the reseeding phase. Alice rates new documents that are not in the gauge set. If enough new users rate these documents, the new documents can be used as a new gauge set. Having dynamic data in the collaborative filter is important; recommendations and predictions of a small, static dataset are subject to attack.

Now we turn to the question of predicting ratings.

4 Using Singular Value Decomposition in Collaborative Filtering

Singular Value Decomposition (SVD) is a numerical method for doing collaborative filtering that separates user ratings by different features. A feature is an abstract notion that falls implicitly out of the decomposition; features require no special annotations in advance. Often, but not always, an abstract feature in the SVD corresponds to a real-world property of the item being rated. For example, one property of jokes is word-play. Users who find word-play humorous might prefer puns, and the word-play property might correspond to a particular feature in the decomposition. For the purposes of this experiment, the real properties corresponding to the features are irrelevant.

If the ratings matrix A holds ratings of users for documents, we can use SVD to decompose A: $A = USV^T$. A_{ij} is user i's rating of document j. It is useful to think of a row in U as a user's response vector, where U_{ik} is user i's response to feature k. S is the matrix of feature weights, or how important a feature is

in determining the rating. S is 0 everywhere but the diagonal, where S_{kk} is the weight of feature k. V represents the amount of each feature in each document, so V_{jk} is the amount of feature k in document j. Since $S_{i \neq j} = 0$,

$$A_{ij} = \sum_k U_{ik} S_{kk} V_{jk}$$

Because of the way the SVD is computed, the first entry in S is the largest, so $S_{11} \gg S_{22} \gg \ldots \gg S_{nn}$. From this, we can estimate the ratings for a new user if we know that user's rating of the first document, R_1. From the previous equation, ignoring the new user's responses to other features, we have

$$R_1 = U_1 S_{11} V_{11}$$

We can then solve for U_1, the user's response to feature 1, and use that to estimate the ratings of other documents. As more ratings are known, the more feature responses we can estimate, and the more accurate the predictions become.

4.1 Measuring Error

The Mean Absolute Error is the error metric used most often in collaborative filtering literature. Let c be the number of jokes rated, p_{ij} be the prediction of user i's rating of joke j, and r_{ij} be the actual rating. Then MAE for user i is

$$MAE = \frac{1}{c} \sum_{j=1}^{c} |r_{ij} - p_{ij}|$$

It is useful to normalize this metric to the range of possible ratings, $[r_{max}, r_{min}]$ [7].

$$NMAE = \frac{MAE}{r_{max} - r_{min}}$$

4.2 Neighbors

Nearest-neighbor algorithms are commonly used to improve predictions using SVD. One way to measure how similar two user preferences are is to measure the distance between their preference vectors. Two users A and B are close if the cosine between their preference vectors is close to 1.

$$\cos\theta = \frac{A \cdot B}{\|A\| \|B\|}$$

One immediate problem with using nearest-neighbor algorithms in a CAPTCHA is that an adversary has fewer data points to guess in order to cheat successfully. We discuss security problems in more detail in Section 6.

5 Experiments with Collaborative Filtering CAPTCHAs

We have presented a class of CAPTCHAs based on collaborative filtering and shown how to implement them. In this section, we conduct two experiments and analyze the security requirements of collaborative filtering CAPTCHAs based on the results.

5.1 A Joke-Based Collaborative Filtering CAPTCHA

We chose to prototype a proof-of-concept collaborative filtering CAPTCHA based on jokes. The subject of jokes was chosen merely for convenience because a large dataset of joke ratings is publically available [6].

While the choice of jokes as the basis for our collaborative filter allowed us to prototype our system quickly, a collaborative filter based on jokes also suffers from a number of flaws. Jokes are often culturally biased, they are hard to generate by computer, and some jokes are offensive. As we discuss below, despite these drawbacks, the use of a joke-based collaborative filter did produce some interesting results and suggests that collaborative filtering deserves further research as an approach for building CAPTCHAs.

There are several possible approaches using jokes to build a CAPTCHA:

- Pick the best joke from a small set.
- Pick the worst joke from a small set.
- Rate the joke.

The third approach, rating the joke, is the most useful because it is the most general — the other two can be implemented based on the rating. If the user's assessment of the joke corresponds to the opinions of previous (human) users, the user passes the challenge. We can then optionally ask the user to rate new jokes and update the collaborative filter. Because this is a proof-of-concept CAPTCHA and because, to the best of our knowledge, this is the first collaborative filtering CAPTCHA, this experiment concentrated mainly on the accuracy of the collaborative filter.

The Jester project The Jester project is a recommender system for jokes [6]. 24953 users in the system rated the same 10 jokes, or *gauge set*, on a scale from -10 (not funny) to 10 (funniest). Although up to 100 jokes were rated, we used only ratings from jokes in the gauge set in this experiment because the gauge set is dense.

To mitigate attacks on the collaborative filter, the data that users rate must fulfill two requirements:

- It must be large or renewable. The Jester system uses 100 jokes. It is possible to compose more jokes, however. If the data is too small, an adversary could simply use a human to rate all the jokes and "replay" known human answers.

– It must be uniformly distributed in quality. If the data does not follow a uniform distribution, an adversary could simply guess the most frequently-occurring rating.

Because neither of these requirements have an effect on the accuracy of predictions for legitimate users, we can disregard them for the purposes of predicting ratings.

5.2 Experiments with the Jester Dataset

In this experiment, we use ratings from the first eight jokes in the gauge set to predict ratings for the last two. We chose at random 100 users from 24953 to use as the training set. This training data was used to compute the feature weight and document matrices (S and V) in the SVD. Recall that S gives the weight of each feature, and V gives the amount of each feature in each document or joke. Because the training data is dense, small and fixed in this experiment, we avoided the problems of *sparsity* and *scalability* described in Section 2. Polluted data will always be a challenge in collaborative filtering CAPTCHAs — there is a trade-off between resistance to adversaries and unfairly failing users with unconventional preferences.

For the other 24853 users (the test data), we used the S and V matrices to predict the ratings of the last two jokes based on the ratings of the first 8. Additionally, all the ratings were normalized linearly to fall between 0 and 1.

Because initial accuracy of the CAPTCHA in distinguishing humans from machines does not depend on the third phase (*reseeding*) in a collaborative filtering CAPTCHA, reseeding is unimplemented.

Results The results of using all the training data without nearest neighbor algorithms to predict ratings for new users are shown in Figures 1 and 2. Figure 1 is a histogram of the cosine between the predicted ratings and the actual ratings for the last two jokes in the gauge set. Figure 2 shows the histogram of the NMAE for the predictions. The NMAE for a random prediction (distributed uniformly over the range) is 0.333 [7]. The NMAE using all the training data for predictions is 0.45, even higher than the expected NMAE for a random prediction.

Unfortunately, using the SVD decomposition of all 100 users in the training data was too inaccurate. Because there is much variation in joke preferences, the predictions were not good using all of the training data.

To improve predictions, we used only the ratings from the 10 nearest neighbors. For each user in the test set, we compared the user ratings for the first 8 jokes to all of the training data, using the cosine as the metric to find the 10 nearest neighbors. The SVD on the 10 nearest neighbors was used to predict the user's ratings for the remaining 2 jokes in the gauge set. Figures 4 and 3 show the results with nearest neighbor.

Figures 4 and 3 show that using nearest neighbors improves the predictions significantly. The NMAE with nearest-neighbor is 0.34, approximately the same

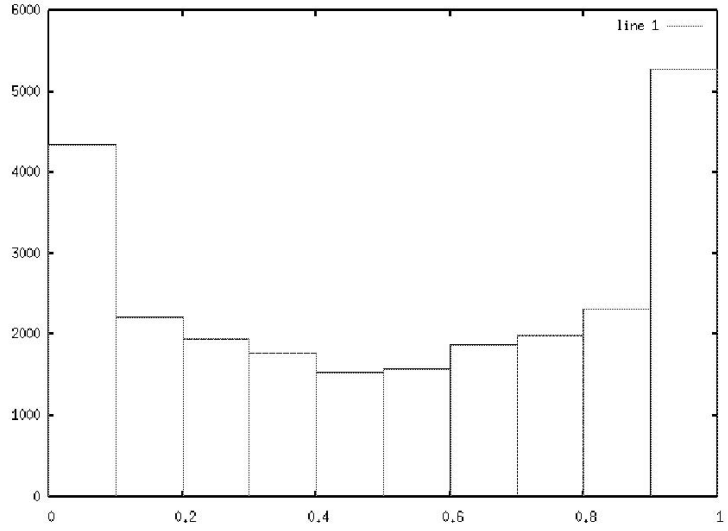

Fig. 1. A histogram of absolute values of cosines between predicted and actual ratings using SVD without nearest neighbor.

as the expected NMAE for random predictions. The NMAE is 0.187 using Eigentaste, the collaborative filtering algorithm developed by the designers of the Jester system [7]. The Eigentaste system uses Principal Component Analysis (PCA) in lieu of SVD. PCA allows dimensionality reduction for fast offline clustering of user preferences. Because this is a proof-of-concept CAPTCHA and the training data is small, PCA is unimplemented.

5.3 Visual Art and Emotions

In this section we describe a speculative collaborative filtering CAPTCHA that requires humans to specify emotions evoked by visual images. One measure for comparing emotions is the Russell circumplex model of affect, illustrated in Figure 5 [13]. The two principal axes are *excitement* and *pleasure*. Russell claimed these axes are orthogonal. For example, the emotion *distress* implies high excitement and displeasure, and so *distress* falls in the upper left quadrant of the model. However, low excitement and displeasure correspond to *depression* in the lower left quadrant. Emotions that are close to each other on the model are perceptually similar to humans, and vice versa. Opposite, or most dissimilar, emotions are diagonal to each other on the model [12].

To devise a CAPTCHA, we can require the user to rate the emotions of images as before and compare the predicted rating to the actual rating. However, SVD assumes a linear scale, not a circular one. As a first step, we can simply use the average rating (or emotion) to predict the rating of a new user, treating emotions as vectors on the unit circle.

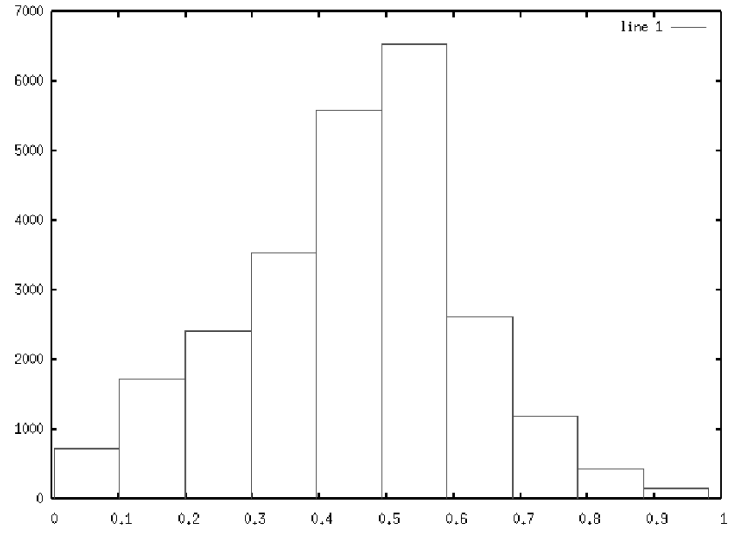

Fig. 2. A histogram of NMAE using SVD without nearest neighbor.

In a small experiment, we chose seven images from an online art gallery, and three random art images. A random art image is simply the result of coloring a random expression in two variables, where the color at a point (x, y) is determined by the value of the expression at that coordinate, as illustrated in Figure 6. Random art is a convenient source of input data to the collaborative filter because it is easy to generate.

Four test subjects chose emotions that best corresponded to the image. However, many of the images were not sufficiently evocative. All of the subjects had difficulty labelling the random art images. Additionally, the four test subjects differed widely on all but one image, where three out of four picked emotions in the same quadrant. Consequently, the average answers were not very meaningful. The result of this small experiment is that the data source that humans rate must be evocative. We also need numerical methods for collaborative filtering that work on higher-dimension scales.

6 Security Against Adversaries

Collaborative filtering CAPTCHAs are predicated on the unpredictability of human opinions. This condition could fail in several scenarios:

– The attacker (human or machine) infiltrates the training data. Then, to pass the CAPTCHA, the attacker can simply rate the gauge set as she did during the training phase. If training users complete the training phase remotely, we can require them to pass another type of CAPTCHA (e.g., an image recognition CAPTCHA) before entering the training phase.

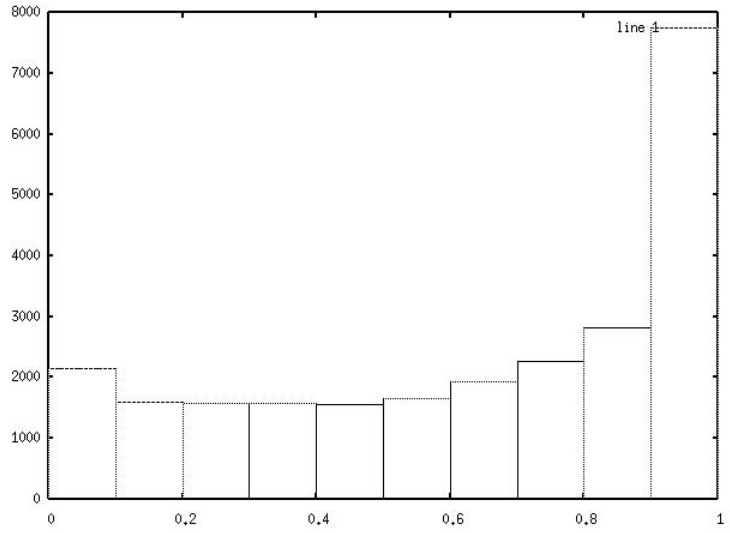

Fig. 3. A histogram of absolute values of cosines between predicted ratings and actual ratings from the SVD predictions with nearest neighbor.

- There is not enough variation in the jokes, i.e., the jokes are consistently no good (or conversely, the jokes are all very funny). In this case, the attacker simply rates each joke the average rating (which is predictable for a small number of jokes) to achieve a lower NMAE than random ratings. To defend against this attack, the ratings of the jokes in the gauge set must be uniformly distributed. The training phase can accomplish this by starting out with a large gauge set and eliminating jokes until the ratings are uniformly distributed. Similarly, in the reseeding phase, only new jokes that preserve the distribution are admitted to the gauge set.
- The attacker guesses a response vector that is consistent with the training data. The attacker then uses the reseeding phase to infiltrate the new gauge set. Of the attacks listed, this is the most insidious. The defense against this attack can include:
  - Introducing new jokes very frequently, so the gauge set is constantly changing.
  - Increasing the size of the gauge set.

From the experiments discussed in the previous section, there are several requirements on the data used for collaborative filtering:

- *Uniform.* Ratings for the data must be uniformly distributed. If the ratings show a bias (as in the case for the Jester dataset), an attacker can use that bias to pass more often than would be expected at random.

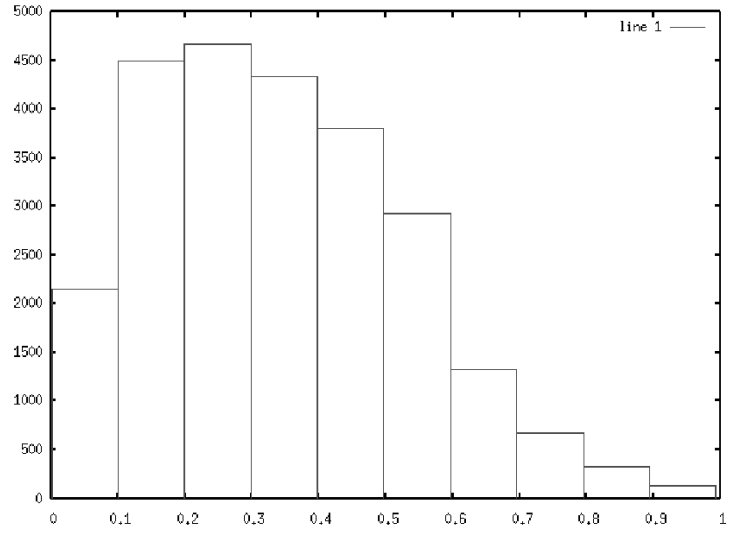

Fig. 4. A histogram of NMAE using nearest neighbor predictions.

- *Evocative.* The data must be sufficiently evocative. In the case of the CAPTCHA outlined in Section 5.3, the emotional effect of the images used was often unclear to the participants, diminishing the meaning of the ratings.
- *Dynamic.* The data must be renewable, or else the same data will appear to an attacker many times. In this case, a correct guess from the attacker will be very valuable, since the attacker can replay it many times.

Another important consideration is that collaborative filtering without using nearest neighbor algorithms failed. The purpose of predicting ratings with the entire training set instead of a subset was to amplify the difficulty of attack. Guessing or controlling the SVD of the entire training set is more difficult than doing the same for a small number of neighbors.

6.1 Exploiting Bias in Ratings

Let X be the user's ratings, Y be the predicted ratings, and $[r_{min}, r_{max}]$ be the range of ratings. In this section, we derive the NMAE between X and Y for three cases:

1. X and Y are uniform random variables. The NMAE for this case is presented in the Jester paper, but we generalize the derivation and show it to be independent of the range [7].
2. X and Y are normally distributed with mean μ and variances σ_x^2 and σ_y^2, respectively. The analysis of this case is summarized from the Jester paper.
3. X is normally distributed with mean μ and variance σ^2, $Y = \mu$. This comparison has not been presented previously.

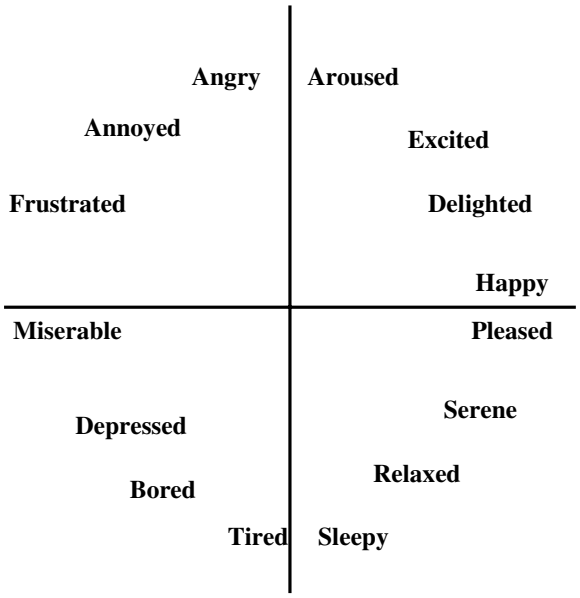

Fig. 5. Russell's circumplex model of emotion.

Uniform distribution Let X and Y be uniform random variables. The probability distribution of the error function $X - Y$ is triangular, ranging from $[r_{min}, r_{max}]$, where $f(x) = x + 20$ for $r_{min} \leq x \leq 0$, and $f(x) = 20 - x$ for $0 \leq x \leq r_{max}$. The probability distribution of the absolute error $|X - Y|$ gives the density function for the MAE. Taking the absolute value folds the function over the y-axis, giving the $f(x) = 20 - x$. Normalizing to integrate to 1, $f(x) = 0.1 - 0.005x$. Taking the integral gives E[MAE] to be

$$\int_{r_{min}}^{r_{max}} f(x)xdx = \int_{r_{min}}^{r_{max}} 0.1x - 0.005x^2 dx = \frac{r_{max} - r_{min}}{3}$$

Normalizing the MAE to the range gives NMAE=0.333, as expected.

Normal distribution Let X and Y be normally distributed random variables with mean μ and variances σ_x^2 and σ_y^2, respectively We can use the moment-generating function to model their difference [8]:

$$M_{X-Y}(t) = M_X(t)M_{-Y}(t) = e^{\frac{1}{2}\sigma_x^2 t^2 + \mu t} e^{\frac{1}{2}\sigma_y^2 t^2 - \mu t} = e^{\frac{1}{2}(\sigma_x^2 + \sigma_y^2)t^2}$$

Thus, the difference is also a normal distribution with mean 0 and variance $\sigma_x^2 + \sigma_y^2$. The density function for the MAE $|X - Y|$ is then

$$f(x) = \frac{2}{\sqrt{2\pi(\sigma_x^2 + \sigma_y^2)}} e^{-x^2/(2(\sigma_x^2 + \sigma_y^2))}$$

Fig. 6. A random art image.

Suppose $\sigma_x = \sigma_y = \sigma$. Then E[MAE] is

$$\int_0^\infty \frac{1}{\sigma\sqrt{\pi}} e^{-x^2/4\sigma^2} x dx = \frac{2\sigma}{\sqrt{\pi}}$$

Normalizing to the range gives an NMAE of $\frac{2\sigma}{\sqrt{\pi}(r_{max}-r_{min})}$.

Normal X, constant Y Let X be normally distributed with variance σ^2 and mean μ, and $Y = \mu$. Then the density function is exactly the standard normal distribution with mean 0 and variance σ^2, and the expected MAE is

$$\int_0^\infty \frac{2}{\sqrt{2\pi}\sigma} e^{-x^2/2\sigma^2} x dx = \frac{\sqrt{2}\sigma}{\sqrt{\pi}}$$

Normalizing to the range gives an NMAE of $\frac{\sqrt{2}\sigma}{\sqrt{\pi}(r_{max}-r_{min})}$. In the Jester data set, the average standard deviation is $\sigma \approx 5$, and the range is 20. Table 1 below summarizes the NMAE for different prediction models.

From Table 1, we can see that guessing the mean is almost as good a predictor as SVD. The cost of this attack is finding the mean. The adversary can do this by using a human to rate many documents, and estimating the mean from that distribution. The number of ratings needed to make a good estimate of the mean depending on the variance of the ratings. To be 90% confident that the true mean is within 3% of the sample mean in the Jester data set, the attacker would need to rate about 187 documents with $\sigma = 5$ and a range of 20:

Table 1. Normalized Mean Absolute Error (NMAE) for different prediction models.

Distribution	E[NMAE]	Jester NMAE
Uniform random	0.33	0.33
Normal	$\frac{2}{\sqrt{\pi}(r_{max}-r_{min})}\sigma$	0.28
Uniform mean	$\frac{\sqrt{2}}{\sqrt{\pi}(r_{max}-r_{min})}\sigma$	0.20
Eigentaste	—	0.187
SVD without clustering	—	0.45
SVD with clustering	—	0.34

$$n = \left(\frac{z_{\alpha/2}\sigma}{.03(r_{max} - r_{min})}\right)^2 = 187$$

This attack is simple and effective. To prevent it, a combination of a uniform distribution of ratings and better predictive algorithms is required.

7 Related Work

The ESP Game A related CAPTCHA-like scheme is the ESP Game developed at CMU [17]. The ESP Game requires two simultaneous users to label 15 images identically within 2 minutes. Moreover, there is a set of "taboo" words that both players are forbidden to use. The ESP Game is not described as a CAPTCHA. One approach to build a CAPTCHA out of the ESP Game would be to accept both players as human if they won the game.

The ESP Game is like a collaborative filtering CAPTCHA in that user data is used to grade the CAPTCHA; however, it requires online interaction, perfect matching between players, and only one other user's data is used to grade the CAPTCHA. Because the ESP Game was not designed as CAPTCHA, it is not surprising that using it as a CAPTCHA would result in a number of problems. These problems include:

- Latency. The ESP Game requires online interaction, so multiple players must simultaneously play the game. In addition, the time to take the ESP game depends on other players, who may be behind a slow network connection or malicious. A collaborative filtering CAPTCHA does not require online interaction with other human users.
- Collusion. Only two colluding players are necessary to fool the system. An adversary can automatically enter the game multiple times until she is paired with herself; then, winning the game is trivially easy. Furthermore, such an attack leads to pollution of the answer database, which is used to label images correctly in an image recognition CAPTCHA. In a collaborative filtering scheme, an adversary must work with her nearest neighbors.

– Unfairness. An adversary can automatically cause a legitimate player to lose by simply entering nonsense answers. In a collaborative filtering scheme, such an adversary would simply fail the CAPTCHA.

A more general approach to collaborative filtering CAPTCHAs sidesteps some of these problems.

Turing Game The Turing Game designed by Berman and Bruckman is one such test, though it is not a CAPTCHA because it is not automated. In the Turing game, the players are separated into the panelists and the audience [1]. The panelists pretend to be members of a particular group (such as women), and the audience of diverse gender asks questions of the panel. After the questioning, audience members vote on who is telling the truth.

SparkLife SparkLife (`community.sparknotes.com`) takes a different approach. It asks a series of fixed, multiple choice questions to determine attributes such as gender, intelligence, all-American-ness, stress level, and greed [15]. The algorithms used for analyzing the answers are not publically known.

8 Discussion

We have proposed a framework for collaborative filtering CAPTCHAs and performed a preliminary security analysis of attack models on the filter. We have shown that collaborative filtering CAPTCHAs require nearest neighbor algorithms to be useful. We have proposed a scheme for updating the collaborative filter to resist attacks and discussed security considerations for the data used in the filter.

The results of the experiments are inconclusive — however, they indicate that collaborative filtering CAPTCHAs are worthy of further investigation.

Open problems include:

– Finding an automatically renewable source of data that users can rate. Jokes must be conceived by humans, for example, but random art images are easy for machines to generate. The problem with random art images, however, is that they may not be sufficiently evocative (Section 5.3).
– Specialized collaborative filtering CAPTCHAs that are targeted at a specific demographic or group of people. Specialized knowledge could aid collaborative filtering CAPTCHAs. For example, a particular dataset (e.g., jokes) elicits different responses from different personality types or demographics. An affinity for puns might indicate linguists, or lingo-philes. A CAPTCHA based on movie data (e.g., user ratings and genre information from the MovieLens project) could target movie buffs [9].
– Using collaborative filtering to improve data sources for other CAPTCHAs. Image recognition CAPTCHAs require a human user to recognize images [2, 4, 16]. Image recognition CAPTCHAs have the problem of *mislabelling*: images in the database are indexed under meaningless labels [4].

Chew and Tygar describe CAPTCHAs requiring three tasks: naming the image by typing the label, distinguishing images, and identifying the anomalous image out of a set. Because all images are culled from Google's image database, not all of the images are labelled correctly. The mislabelling problem causes humans to fail CAPTCHAs. We can use collaborative filtering to eliminate or reduce poorly labelled images.

References

[1] Joshua Berman and Amy S. Bruckman. The Turing game: Exploring identity in an online environment. *Convergence*, 7(3):83–102, 2001.

[2] Manuel Blum, Luis A. von Ahn, John Langford, and Nick Hopper. The CAPTCHA Project. http://www.captcha.net, November 2000.

[3] Jack Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Fourteenth Conference on Uncertainty in Artificial Intelligence*, 1998.

[4] Monica Chew and J. D. Tygar. Image recognition CAPTCHAs. In *7th Annual Information Security Conference*, pages 268–279, August 2004.

[5] Chrysanthos Dellarocas. Immunizing online reputation reporting systems against unfair ratings and discriminatory behavior. In *2nd ACM Conference on Electronic Commerce*, 2000.

[6] Ken Goldberg, Robert Hennessy, Dhruv Gupta, Chris Perkins, Hiro Narita, and Mark DiGiovanni. Jester. http://shadow.ieor.berkeley.edu/humor, 1999.

[7] Ken Goldberg, Theresa Roeder, Dhruv Gupta, and Chris Perkins. Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval*, 4(2):133–151, 2001.

[8] William Mendenhall and Terry Sincich. *Statistics for Engineering and the Sciences*. Dellen Publishing Company, 3rd edition, 1992.

[9] B. Miller, I. Albert, S.K. Lam, J. Konstan, and J. Riedl. Movielens unplugged: Experiences with a recommender system on four mobile devices, 2003.

[10] Greg Mori and Jitendra Malik. Recognizing objects in adversarial clutter: Breaking a visual CAPTCHA. In *Computer Vision and Pattern Recognition*, 2003.

[11] Gabriel Moy, Nathan Jones, Curt Harkless, and Randall Potter. Distortion estimation techniques in solving visual captchas. In *Computer Vision and Pattern Recognition*, 2004.

[12] R. T. Ross. A statistic for circular scales. *Journal of Educational Psychology*, 29:384–389, 1938.

[13] J.A. Russell. A circumplex model of affect. *Journal of Personality and Social Psychology*, 39:1161–1178, 1980.

[14] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *The 10th International World Wide Web Conference*, 2001.

[15] SparkNotes. http://community.sparknotes.com, 1998–2004.

[16] Luis von Ahn, Manuel Blum, and John Langford. Telling apart humans and computers automatically. *Communications of the ACM*, 47(2):57–60, 2004.

[17] Luis von Ahn et al. The ESP game. http://www.espgame.org, 2004.