# Enhancing Security and Usability from a Human Perspective on the World Wide Web

by
David Lorenzi

A dissertation submitted to the
Graduate School-Newark
Rutgers, The State University of New Jersey
In partial fulfillment of the requirements
For the degree of
Doctor of Philosophy
Graduate Program in
Management Sciences & Information Systems - Information Technology
Written under the direction of
Dr. Jaideep Vaidya
And approved by

Newark - New Jersey, USA

October 2016

# Abstract of the Dissertation

**Title: Enhancing Security and Usability from a Human Perspective on the World Wide Web**

**By: David Lorenzi**

**Dissertation Director:**

**Dr. Jaideep Vaidya**

Completely Automated Public Turing test to tell Computers and Humans Apart or CAPTCHA, play a pivotal role in governing access to resources made available on the World Wide Web. In an age where online resources can be exploited by those with the ability to leverage automation to utilize these resources outside of their intended use cases, CAPTCHAs provide a method for testing if a particular user who wishes to conduct an activity or consume a resource is a human or a bot. CAPTCHAs achieve this security through the use of a hard AI problem as a challenge response to a request for resources - specifically a task that is easy for a human to solve quickly but difficult or impossible for a computer to solve in the same amount of time. When used in conjunction with other methods of online access and form control, CAPTCHAs can help secure the Web from automated exploitation, bots, spam, and other such abuses. CAPTCHAs are a perpetually evolving area of research, due in part to their function as a security method and consequently are forever embroiled in an arms race between blackhats developing new attacks against best-of-breed CAPTCHAs currently deployed and whitehats trying to defend their resources against these attacks with new styles of CAPTCHA and techniques to defeat attack methods. This dissertation focuses primarily on Image Recognition CAPTCHAs or IRCs, as the CAPTCHA of choice to provide reasonable security for the Web while maintaining acceptable usability for humans.

Two attack methods researched for defeating IRC challenges are discussed, one which focuses on outright attempts at image classification through the use of a specialized neural network (HTMs), and another which utilizes web services to exploit metadata associated with images to circumvent performing the image classification task and still correctly answer the challenge. Two defensive methods researched and developed for securing IRC challenges against these types of attacks are also discussed. The first method focuses on the addition of noise to an image to prevent an attacker from being able to effectively leverage web services to gather metadata and other useful data typically needed by computer vision algorithms, such as structure, patterns, or colors from the image. The second is designed to stop computer vision (CV) algorithms and web services from being able to extract contextual information and metadata from an image through the application of a series of image filters, yet allow a human to still discern this information.

User studies are provided for both defensive methods to test the real world usability of the method in practice on an IRC, as well as the CAPTCHA design style they were implemented in, of which we provide a number of variations. An in-depth discussion on CAPTCHA theory and design considerations as well as an overview of some new, original CAPTCHA designs are presented for the reader. Analysis and speculation for the future direction CAPTCHAs could develop is provided as well. Finally, coverage of the design and implementation of a scalable and robust IRC that relies on a human being able to detect contextual information from an image to solve the challenge is demonstrated as the culmination of this body of research.

# Preface

## Dissertation Committee Members

- Dr. Nabil Adam, Rutgers University
- Dr. Vijayalakshmi Atluri, Rutgers University
- Dr. Jaideep Vaidya, Rutgers University
- Dr. Shamik Sural, Indian Institute of Technology Kharagpur

# Acknowledgments

First, I would like to thank my dad for providing the foundation to enable all of this work to be possible. I am truly grateful for the opportunities you have provided for me in life and the sacrifices you made to make them possible. Second, I would like to thank Dr. Jaideep Vaidya, for being my guide and mentor on this journey through research. I will cherish the time we spent contemplating challenging avenues of research for a lifetime. I will forever appreciate the chance you took on a young masters student with dreams of research. Third, I would like to thank my committee members for all of the quality input and guidance you all have provided me over the years - in particular, I very much appreciated the teaching assistant position and dissertation fellowships as they provided me with the chance to experience teaching at a university level and conducting original research. A special thanks to Dr. Nabil Adam and Dr. Soon ae Chun for the research opportunities for me within the CIMIC group. It was truly a pleasure to work and study with the CIMIC group, and I hope to continue to contribute to research into the future. I would also like to thank assistant dean Goncalo Filipe for his tireless administrative efforts on my behalf, the work you put in does not go unnoticed or unappreciated. Finally, a special thanks to all of my friends I made throughout my journey, you all gave me the support and encouragement I needed to continue my studies and see them to completion.

**Special Thanks:** Emre Uzun, the Naranjos, Raymond Wong, Amanda Dios, Max Luebbe, and 1210 Krew

# Table of Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

With the pervasiveness of the Internet and the World Wide Web, it is increasingly advantageous for attackers to begin abusing web services and online forms for personal and financial gain. Web administrators need a way to curtail or stop the bots' abuse, while not restricting their human users from accessing the resources they provide. Thus, the CAPTCHA was born as a method to discern between a bot and human user online. The CAPTCHA, which stands for Completely Automated Public Turing test to tell Computers and Humans Apart, was invented to discern between a bot and human user online. CAPTCHAs are reverse Turing tests administered by computers designed to keep bots from abusing web services and online forms made for human users. CAPTCHAs rely on hard AI problems to provide the challenge question asked to the user (human or bot). This ensures that the challenge question is one that is difficult for a computer to perform with a high degree of success, yet still remains easy for a human to perform quickly [3]. These types of questions provide the foundation for a vast majority of CAPTCHAs. Fortunately, CAPTCHAs have been reasonably successful in stopping a majority of bot abuse at the minor inconvenience of users.

CAPTCHAs are now ubiquitously found on the web to ensure that the entity interacting with a website is indeed human. While CAPTCHAs ensure that abuse of online forms is reduced, web users are forced to suffer through increasingly convoluted and unfriendly CAPTCHAs that negatively impact their user experience. Unfortunately, CAPTCHAs have become a "necessary evil" for online services, especially ones that are designed for public use or are free in terms of usage cost and/or registration to the end user. CAPTCHAs are "evil," in a sense, because they impede service for a user by requiring them to solve a challenge to continue; in essence, they take up system resources (CPU time, bandwidth etc.) that could be spent on enhancing the service they are protecting. Users frequently report becoming angry or not performing an action, such as posting a comment, if they are forced to solve a CAPTCHA to perform that action. However, blackhats and bots continue pumping out spam in all forms: emails, junk comments, link spam etc., whether or not form controls are implemented.

The real task for any attacker is attempting to either directly solve the hard AI problem put forth by the CAPTCHA, or find other ways to "work around" the problem, such as: side channel attacks, mechanical turks, remote farming, exploitable implementation flaws etc. However, many spambots simply look for low hanging fruit, that is, online forms that are unsecured or have easily circumvented protections e.g. ones that have tools that break them automatically. When viewed from an economic perspective (often this is the motivation for a majority of spam), this behavior on behalf of the spambot is logical, as it is not as cost effective expending effort to break into hardened targets. A vast majority of cybercriminals would not bother developing advanced attacks against image CAPTCHAs that would presumably require sophisticated computer vision techniques to attempt to break.

Text-based CAPTCHAs are the most common implementation in use, due to their scalability, robustness, and ease of implementation. ReCAPTCHA, a formerly text-based implementation owned and operated by Google, serves as the de facto standard to which all other text-based CAPTCHAs can be compared. Note that as of 2016, the text based CAPTCHA challenges from ReCAPTCHA are now considered, “legacy CAPTCHA” and the reason for this will be detailed later. While the difficulty (for users and bots alike) of text based implementations can be scaled according to how willing the CAPTCHA administrator is to making the CAPTCHA vulnerable to various attacks, it invariably leads to the age old problem of making tradeoffs between usability and security. However, given the continued prevalence of text based CAPTCHA, many techniques have been developed to break them. As a result, alternative methods of form control and human verification have been sought for by the security research community. Among the several different modalities that have been explored, image based CAPTCHAs have emerged as a plausible alternative for the future, as more suitable for the smartphone/mobile touch-capable environment. However, image CAPTCHAs come with their own set of problems, particularly in terms of scalability – it is hard to find large quantities of labeled/tagged images; and robustness – there is limited variation in the challenge question and vulnerability to single style of attack.

Very recently, an important shift in enterprise scale CAPTCHA implementations occurred – Google has begun to move away from text based CAPTCHAs toward image based CAPTCHAs, as they have discovered that deep-learning neural networks for vision tasks were exceptionally well equipped to handle the unique challenges of text based CAPTCHAs [65, 31]. Despite the best efforts to obfuscate the text with tricks to fool optical character recognition methods and computer vision models (e.g., overlapping characters, boundary color distortions, character warping, additional noise in the form of lines, etc.) the amount of obfuscation required to defeat the deep-learning model reached a level where the CAPTCHA challenges are now too difficult for a majority of humans to solve reliably. The regularity and structure of the characters causes the problem, since at a certain point increasing variation in the structure of the characters also makes them incomprehensible to a human. CAPTCHA designers have since begun to migrate to different forms and styles, and away from text based challenges. While deep-learning neural network based models remain primarily academic and exist only in the hands of a small knowledgeable elite, it will not be long before these tools and techniques reach the computing mainstream, and have already begun emerging into the marketplace in mid-2016. The shift to an image based model presents its own set of obstacles to overcome by both the service provider and the attacker, and soon enough, deep learning models may yet again prove to be powerful enough to overcome previously challenging computer vision related tasks. There is already some evidence that this is the case, with the ability to discern emotion [49] and age [37] from faces, generate image tags based on the content of an image [**?**], and identify multiple objects in complex scenes [53]. Furthermore, emerging neuroscience research has even provided methods to create human–machine interfaces that allow a computer to read a person's mind via an analysis of brain waves to determine the category an image belongs to [50]. It is easy to see there exists the very real possibility that simple cognitive tasks that are easy for a human to perform will become something that can be accomplished by a computer using a number of different algorithms and learning models to affect this outcome. This leads some in the security community to believe a change in direction is needed.

## 1.1 Human Verification - Strong Identity Vs. CAPTCHAs

There has also been a recent shift away from CAPTCHAs toward what is being termed "strong identity" as a method for providing user privileges and access to online resources. Strong identity as an online security concept is rooted in the leveraging of the verifiable "real world". The intention behind this idea is that using multiple forms of identification to get a better sense of who the user is and what their behavior will be like when using the service can be useful from a security perspective [28].

For example, when signing up for a new account with a service, if the user links their facebook profile, twitter, linkedin, etc., and allow them to be scanned by an algorithm that examines user behavior and performs cross validation of content, where the user will not have to solve a CAPTCHA or wait for an administrator to approve the creation of the account [14]. Social networking sites stand out in particular as service providers who are in favor of this security model [33]. Strong identity takes the form of online service providers encouraging the user to use their real name and provide truthful, accurate information when creating an account; information that can be cross referenced to "official" information sources (e.g. SSN, home address, telephone number etc.) and is unique to an individual.

Socio-behavioral models [62] that correlate with strong identities are also gaining traction as a new method to identify bot behavior. Behavioral models are based around the idea that a typical "legitimate" user performs a set of actions that are considered by the online service provider to be typical of the use of the services provided. Since bots have a fixed set of actions that are usually designed to maximize some type of outcome for an attacker, their behaviors online become an outlier that the service provider can identify and act upon accordingly.

For example, a typical human user of a social network would add or follow some friends, interact with them by sending messages and comments, share links etc. A bot would take this behavior to the extreme, perhaps attempting to add or follow 500 or more users upon creation of the account and immediately send a large volume of messages to users – behaviors that would require beyond human abilities. Another example of strong identity would be the implementation of two-factor authentication – some online services are providing a unique text message to a given phone number to verify identity, a method that is becoming increasingly popular as hackers breach accounts secured only by traditional passwords.

It is worth noting that strong identity models gain much of their power from traditional human social and legal structures that provide feedback to users in the form of punitive measures for straying from approved behavioral norms and written laws. If abusive accounts can be linked to a real world human that is within the reach of litigation that can be brought by the service provider, this is even better in the eyes of the service provider. Now the disincentive for abusing the service is much greater, as legal action can be brought upon an individual or group abusing the service. However, these methods are somewhat limited in that they completely break down when jurisdiction becomes an issue, (e.g. a foreign sovereign country may choose not to bring charges against a citizen abusing another countries' online services). However, while strong identity models sound great in theory, the use of real world information online can result in negative consequences if compromised (ID Theft, stalking/harassment, loss of reputation etc.) and comes at the expense of anonymity and privacy. Indirectly, CAPTCHAs provide a way to allow users to remain anonymous without having to provide real information to prove their humanity. Additionally, such strong identity approaches may not be suitable for all types of web based applications. Thus, it would not be a stretch to say that if you value privacy online, it is in your best interest to support and develop new CAPTCHAs.

Recent research has also looked into using CAPTCHA generation technology that leverages hard AI problems to provide a graphical replacement for the traditional password (CaRP) [77], as passwords are becoming increasingly insecure online due to various human limitations in remembering unique, long and complex strings of characters and increases in the speed and power of attacks on password databases and systems. It will be interesting to see if graphical based challenge systems can evolve over time to position themselves to provide strong online form and account security by providing usable, scalable and robust challenges to users for identification and authentication tasks.

An argument can be made that the true intention of CAPTCHAs is to deincentivize human greed. We define greed in this case to mean, “The desire for more than the service providers offered fair amount of scarce/finite resources” - whatever that may be e.g. download link, email account, cloud storage space etc. CAPTCHA security is provided by removing the ability to automate this greedy behavior, with the bot acting as an agent – on behalf of the blackhat. While changing human behavior is difficult, it can be controlled to a degree with the right incentives and disincentives in place to produce the desired outcomes. Finally, it must be pointed out that CAPTCHAs should never be used in isolation, they must be a living component of a comprehensive security system designed to prevent abuse of an online service. Methods such as invisible fields and random field names on forms, IP whitelisting/blacklisting, throttling (bandwith use, volume of messages etc.), filtering (words, links, content etc.) as well as combinations of automated and manual account inspection have been used successfully to curb abuse in conjunction with CAPTCHAs [41]. It is important to remember that online form security is a “puzzle,” and all the pieces must work in conjunction to provide the level of protection the service provider desires.

## 1.2 Problem Statements and Contributions

CAPTCHAs are an ever evolving area of research, as their ability to provide security is dependent upon the current state of the art computer vision algorithms, machine learning algorithms, and artificial intelligence in circulation at the time of their emergence onto the World Wide Web. As CAPTCHA styles are defeated by new attacks, the entire landscape must react to the emergence of these new threats. The temporal nature of the state of affairs in regards to attack and defense in CAPTCHA cannot be overstated. For example, during the length of time it took to conduct the research for this dissertation, there has been an active shift away from text based CAPTCHA to image based CAPTCHA (in both research literature and online implementations), especially with the emergence of deep learning neural network models providing extremely high levels of accuracy in solving even the most difficult text based implementations. Google offering ReCAPTCHA as a service to web administrators is a prime example of the shift to IRC, however due to usability issues, Google still offers the traditional text challenge as a legacy option. However, ironically enough, the classic text CAPTCHA is still the most widely used form of CAPTCHA, despite its weaknesses being known. This is most likely due to a number of reasons, namely that developing scalable IRC's that can produce unique challenges at extreme volume of demand in an automated fashion remains a challenging and complex task. Yet even under these circumstances the text CAPTCHA still provides enough protection that web administrators deem them useful and continue to implement as a part of their anti-bot toolbox. The following contributions demonstrate the research path taken which finally lead to the creation of a scalable IRC CAPTCHA and is demonstrative of the temporal nature of CAPTCHA research. Each of these contributions built upon the other and time was required to pass before certain tools and techniques emerged that made certain functions possible that were most likely not possible before. Regrettably, this dissertation is something of an anachronism, as some of the original contributions have been superseded by more advanced techniques (namely the two attacks), but a number of the defensive methods and models remain useful to date.

This dissertation seeks to contribute to the existing body of CAPTCHA research and development through the following original contributions:

1. The development and testing of a new attack on existing IRC implementations using Hierarchical Temporal Memory, a specialized type of neural network that attempts to mimic the mammalian neocortex for the express purpose of image classification and object recognition tasks.

2. The development and testing of a new attack on existing IRC implementations that utilizes web services such as Reverse Image Search (RIS), Automated Linguistic Annotation (ALA), and Image Similarity Search (ISS) that perform image operations to gather metadata relating to images in order to circumvent performing an image identification task, yet allow an attacker to still be able to solve the challenge using the metadata to perform the identification task accurately.

3. The development and testing of a new defense method that can be added to any existing IRC to strengthen it against web services attacks and computer vision attacks while only minimally impacting usability. This is achieved through the application of noise from specialized noise algorithms to the image so that these images fail to return meaningful results to an attacker using online tools and computer vision algorithms.

4. The testing and conducting of a user study of the noise defense method in action through the development of five basic image recognition CAPTCHAs designed to test various aspects of usability inherent to simple IRC's where the central challenge question asked to the user is an image recognition or categorization task.

5. The development of an automated system to generate IRC challenges that will systematically apply noise in increments and test images against web service based attack tools to ensure they meet a security guarantee that RIS nor ISS will return meaningful results (zero matches).

6. The testing and development of a new defense method that can be added to any existing IRC that to strengthen it against web services attacks and computer vision attacks as well as additional advanced image analysis algorithms. This

is achieved through the application of image filters to existing images in order to prevent the extraction of contextual information from the image.

7. The testing and conducting of a user study of a new IRC that is both automateably scalable and robust against attacks from web services, computer vision algorithms, and advanced image analysis algorithms. This new CAPTCHA design asks the user to identify emotions people are expressing on their faces and match them with an emoji that expresses the same emotion. There are also two alternative styles of this basic format that leverage the same concepts yet strive for improved usability.

8. A number of examples of CAPTCHA styles and implementations that use specific hard AI challenges to increase the security provided by the particular CAPTCHA challenge. These examples serve to demonstrate the CAPTCHA design criteria we outline: usability, scalability, and robustness. They also clearly demonstrate the various tradeoffs in security and usability as a consequence of design choices.

## 1.3 Outline

The rest of the dissertation is organized as follows: Section 3 discusses CAPTCHA general design requirements, discusses challenges faced by designers, outlines criteria for designing a quality, secure CAPTCHA system, and provides an overview and evaluation of a number of challenge ideas. Section 4 provides discussion in detail of the attack methods developed, the first with HTMs for image recognition and categorization tasks, and the second using web services to leverage image metadata to solve the challenge without explicitly performing the image recognition/classification task. Details of the performance of the attacks on a selection of IRCs is provided. Section 5 describes in detail the SIGNAC method, our noise addition method for

IRCs to prevent web service and computer vision attacks. Design criteria for a fully automated secure image production system are provided, as well as details of the experiments conducted to determine how much noise and of what type was required to defeat various web service attacks and computer vision attacks. The EmojiTCHA image filter method, which is in a similar vein to the noise addition method and seeks the same outcomes, yet can be achieved with a much more streamlined and less complex system of tools than SIGNAC is also detailed. Section **??** provides coverage of the user studies conducted to test usability of simple image recognition CAPTCHAs utilizing the SIGNAC method, as well as usability testing for EmojiTCHA and the image filter application method. Section 7 provides some ideas for future work that this body of research has identified, as well as some lessons learned over the years of working with CAPTCHA.

# Chapter 2

# Related Work

In this section we will cover a number of works in the literature whose findings and methods have consequences that feed into CAPTCHA analysis and design. Some of these works fall into the computer vision and analysis category, as these papers focus on techniques and algorithms that achieve a particular task that may be used at some point to solve a CAPTCHA or defeat a security mechanism within a CAPTCHA challenge. A section is dedicated to works that use CAPTCHA to achieve other goals or discuss overarching design criteria or critique existing criteria within a CAPTCHA type.

## 2.1 Computer Vision and Machine Learning

Computer vision and machine learning methods drive most of the attacks against CAPTCHAs of all types, as even text based CAPTCHA have a visual component to them (display of the characters). As such, understanding the methods, techniques, and tools that have been developed in these areas of research will go a long way towards helping to fashion both new attacks, as well as new defensive methods to protect CAPTCHAs from automated attack methods. As more of computer vision and machine learning tools become publically available online, their use will increase, and a corresponding rise in attacks against styles vulnerable to them can be expected. A great example of this is the online computer vision tools, provided by CMU [45].

Work by Belongie [5] provides a novel approach to measuring similarity between shapes for the purpose of object recognition. This method is very well suited to the challenges of recognizing characters (digits in particular) that have been warped or distorted by a CAPTCHA, as handwriting differences between people is similar to how image distortions on a CAPTCHA character would look. After a segmentation of the characters in the CAPTCHA occurs, this technique can be used to compare the segmented characters to a template of characters for determining a match based on a similarity score.

Gao [26] provides a new look at utilizing existing hierarchical ML methods to provide accurate classifications in a multiclass classification problem (e.g. for multiple object recognition or scene classification) using a set of binary classifiers in a hierarchical structure. This is similar in style to the HTM algorithm we use in to try to break image classification CAPTCHAs, and follows a similar approach.

Work by Mehta [56] on email spam mirrors work on CAPTCHA in many ways, and the lessons from this experiment illustrate the cat and mouse game played by attackers and designers of CAPTCHA, especially in regards to image analysis. They note that even though current email spam detecting software has been gaining a competitive edge against text based email spam, new advances in spam generation have posed a new challenge: image-based spam. Image based spam is email which includes embedded images containing the spam messages, but in binary format. They analyze the characteristics of image spam to propose two solutions for detecting image-based spam. The first solution, which uses the visual features for classification, offers an accuracy of about 98%. SVMs (Support Vector Machines) are used to train classifiers using judiciously decided color, texture and shape features. The second solution offers a novel approach for near duplication detection in images. It involves clustering of image GMMs (Gaussian Mixture Models) based on the Agglomerative Information Bottleneck (AIB) principle, using Jensen-Shannon divergence (JS) as the distance measure. It is worth noting that similarity/duplication detection is used for matching, as we also leverage this technique for attacks on CAPTCHA.

Work by Chechik [10] presents OASIS, an Online Algorithm for Scalable Image Similarity learning that learns a bilinear similarity measure over sparse representations. OASIS is an online dual approach using the passive-aggressive family of learning algorithms with a large margin criterion and an efficient hinge loss cost. Our experiments show that OASIS is both fast and accurate at a wide range of scales: for a data set with thousands of images, it achieves better results than existing state-of-the-art methods, while being an order of magnitude faster. For large, web scale, data sets, OASIS can be trained on more than two million images from 150K text queries within 3 days on a single CPU. On this large scale data set, human evaluations showed that 35% of the ten nearest neighbors of a given test image, as found by OASIS, were semantically relevant to that image. This suggests that query independent similarity could be accurately learned even for large scale data sets that could not be handled before. This type of machine learning research provides the foundation for image similarity search engines and reverse image search, which we demonstrate can be used to attack CAPTCHAs that use images gathered from an image index online.

Shrivastava [67] has provided a method to find visually similar images even if they appear quite different at the raw pixel level. This task is particularly important for matching images across visual domains, such as photos taken over different seasons or lighting conditions, paintings, hand-drawn sketches, etc. They achieve this with a method that estimates the relative importance of different features in a query image based on the notion of ”data-driven uniqueness” using standard tools from discriminative object detection in a novel way, yielding a generic approach that does not depend on a particular image representation or a specific visual domain. It demonstrates good performance on a number of difficult cross-domain visual tasks e.g., matching paintings or sketches to real photographs.

Work by Tsai et. al. [69] is particularly important to the evolution of image similarity search being able to return a text string "guess" as to what is being depicted in an image. This work addresses the problem of large-scale annotation of web images. Their approach is based on the concept of visual synset, which is an organization of images which are visually-similar and semantically-related. Each visual synset represents a single prototypical visual concept, and has an associated set of weighted annotations. Linear SVMs are utilized to predict the visual synset membership for unseen image examples, and a weighted voting rule is used to construct a ranked list of predicted annotations from a set of visual synsets. They demonstrate that visual synsets lead to better performance than standard methods on a new annotation database containing more than 200 million images and 300 thousand annotations, which is the largest ever reported. It is worth noting that this research was sponsored by Google, who has strong image search and annotation tools available through their search engine, both of which we use in attacks against IRCs as well as test against for defensive methods.

Datta et. al. [16] have implemented some of the pioneering works in CBIR and automatic image annotation. This paper in particular provides an excellent overview of the past, present and future of CBIR and automated image annotation. In this article, they survey almost 300 key theoretical and empirical contributions in the current decade related to image retrieval and automatic image annotation, and in the process discuss the spawning of related subfields. We also discuss significant challenges involved in the adaptation of existing image retrieval techniques to build systems that can be useful in the real world. While this paper was published in 2008, many of the predictions they have made for the future have come true, and now in 2016, automated image tagging and scene recognition tools are available online that have a shocking degree of accuracy unthinkable almost slightly under a decade ago.

Jing's work on[43] the task of identifying "authority" nodes on an inferred visual similarity graph is important to the foundational ideas behind image similarity search engines. They propose an algorithm to analyze the visual link structure that can be created among a group of images. Through an iterative procedure based on the PageRank computation, a numerical weight is assigned to each image; this measures its relative importance to the other images being considered. The incorporation of visual signals in this process differs from the majority of large-scale commercial-search engines in use today, as they often solely rely on the text clues of the pages in which images are embedded to rank images, and often entirely ignore the content of the images themselves as a ranking signal. This is followed up in 2012 by the work on Google image swirl [44]. Google Image Swirl demonstrates the first large-scale image browsing system applied to 200,000 popular queries which utilizes image content to organize image search results. Given a query, the system extracts image content features such as color, shape, local features, face signatures and metadata from up to 1000 image results, and hierarchically clusters them to form an exemplar tree. A dynamic web-based user interface allows the user to navigate this hierarchy, allowing fast and interactive browsing. The exemplars of each cluster provide a comprehensive visual overview of the query results, and allow the user to quickly navigate to the images of interest. It is easy to see where this leads, as online image search has become a powerful tool to provide context and analysis for images in an easy to access fashion. The ramifications for this to IRCs must be considered if robust challenges are to be created.

Mitra et.al.[57] work on emerging images provides some insight into the human-computer gap that is often exploited to create strong IRCs. Emergence refers to the unique human ability to aggregate information from seemingly meaningless pieces, and to perceive a whole that is meaningful. This special skill of humans can constitute an effective scheme to tell humans and machines apart. This paper presents a synthesis technique to generate images of 3D objects that are detectable by humans, but difficult for an automatic algorithm to recognize. The technique allows generating an infinite number of images with emerging figures. The algorithm is designed so that locally the synthesized images divulge little useful information or cues to assist any segmentation or recognition procedure. As a consequence of this, the computer vision algorithms are incapable of effectively processing such images. However, when a human observer is presented with an emergence image, synthesized using an object they are familiar with, the figure emerges when observed as a whole. The difficulty level of perceiving the emergence effect can be controlled through a limited set of parameters. They note that a procedure that synthesizes emergence images can be an effective tool for exploring and understanding the factors affecting computer vision techniques. Jian et. al. continue exploring this path by emerging images synthesis from photographs [42]. They are able to demonstrate a technique that uses a base photograph to create an emerged image using some anti-computer vision techniques. Emerging images represents a hopeful new avenue for IRCs to take into the future.

## 2.2 Other CAPTCHA Works

This section covers a number of papers that are important to CAPTCHA work, but may not be directly involved in attacks on CAPTCHA and defensive designs of challenges, but still impact them in a meaningful way. They focus on human computer interaction and implementation issues, as well as leveraging CAPTCHA for different purposes beyond a simple form control method. Some literature that provides how CAPTCHA design has an aspect of cultural dynamics or provides a human perspective on solving hard AI problems.

Datta et al. continue their work on image processing and CBIR to help design stronger CAPTCHAs by exploiting the human machine gap in image recognition for designing CAPTCHAs[17] they explore the exploitation of this limitation for potentially preventing automated network attacks. While undistorted natural images have been shown to be algorithmically recognizable and searchable by content to moderate levels, controlled distortions of specific types and strengths can potentially make machine recognition harder without affecting human recognition. This difference in recognizability makes it a promising candidate for automated Turing tests which can differentiate humans from machines. They empirically study the application of controlled distortions of varying nature and strength, and their effect on human and machine recognizability. While human recognizability is measured on the basis of an extensive user study, machine recognizability is based on memory-based content-based image retrieval (CBIR) and matching algorithms. A detailed description of our experimental image CAPTCHA system, IMAGINATION, that uses systematic distortions at its core. A significant research topic within signal analysis, CBIR is actually conceived here as a tool for an adversary, so as to help us design more foolproof image CAPTCHAs. Using the idea of system duality, Faymonville [22] introduces an open labeling platform for Computer Vision researchers based on CAPTCHAs, creating as a byproduct labeled image data sets while supporting web security. For the two different tasks of annotation and detection, they explore usability issues and discuss system sustainability issues in the context of a broader ecosystem for the platform. Yan et. al. provide an in depth review of the idea of CAPTCHA robustness, from a security engineering perspective [76]. However, the robustness of CAPTCHAs has so far been studied mainly just in communities such as computer vision, and document analysis and recognition. This paper motivates a security engineering perspective of the robustness of CAPTCHAs. Specifically, they demonstrate that a number of CAPTCHAs that appeared to be secure, including schemes widely deployed by Microsoft, Yahoo and Google and some other less well-known ones, could be broken with a high success rate with simple but novel attacks. In contrast to earlier work that relied on sophisticated computer vision algorithms, our attacks exploited critical design errors that we discovered in each scheme (which are text based). The main lesson is that security engineering expertise and experience, in particular adversarial thinking skills, can make a unique and significant contribution to the improvement of the robustness of CAPTCHAs.

Fritsch [24] develop an online attack that uses a fuzzy image recognition algorithm to process the difference in pictures of nature from pictures that are not nature (the challenge present in the HumanAuth CAPTCHA). To evaluate the attack they implemented a publicly available tool, which delivers promising results for the HumanAuth CAPTCHA and others of a similar style and challenge task. Based upon their findings they propose several techniques for improving future versions of image recognition CAPTCHAs, however most of these methods have been superseded by advanced image processing tools e.g. TinEye. This work represents a direct attack on the hard AI problem posed by the challenge. However, implementation flaws can also provide meaningful ways to attack CAPTCHA as well. Castro et al. propose a side channel attack on the humanauth CAPTCHA [34] which represents a significant shortcut to the intended attacking path, as it is not based in any advance in the state of the art on the field of image recognition. After analyzing the HumanAuth image database with a new approach based on statistical analysis and machine learning, they concluded that it cannot fulfill the security objectives intended by its authors. Then, they analyze which of the studied parameters for the image files seem to disclose the most valuable information for helping in correct classification. They also analyze if the image watermarking algorithm presented by the HumanAuth authors is able to counter the effect of this new attack. The attack represents a completely new approach to breaking image labeling CAPTCHAs, and can be applied to many of the currently proposed schemes. Lastly, they also investigate some measures that could be used to increase the security of image labeling CAPTCHAs as HumanAuth, but conclude no easy solutions are at hand.

Bursztein et. al. present their comprehensive work on how good are humans at solving CAPTCHAs [8]. This is the first large scale evaluation of CAPTCHAs in the literature from the human perspective, with the goal of assessing how much friction CAPTCHAs present to the average user. For the purpose of this study they have asked workers from Amazons Mechanical Turk and an underground captcha breaking service to solve more than 318,000 CAPTCHAs issued from the 21 most popular captcha schemes (13 images schemes and 8 audio scheme). Analysis of the resulting data reveals that CAPTCHAs are often difficult for humans, with audio CAPTCHAs being particularly problematic. They also discovered some demographic trends indicating, for example, that non-native speakers of English are slower in general and less accurate on English-centric captcha schemes. Evidence from a weeks worth of eBay CAPTCHAs (14,000,000 samples) suggests that the solving accuracies found in the study are close to real-world values, and that improving audio CAPTCHAs should become a priority, as nearly 1% of all CAPTCHAs are delivered as audio rather than images. Finally the study also reveals that it is more effective for an attacker to use Mechanical Turk to solve CAPTCHAs than an underground service. In a similar vein, Fidas [23] created a questionnaire-based survey combined with a real usage scenario of a native-language CAPTCHA mechanism conducted an experiment in order to investigate several aspects that affect end-user perceptions related to the quality of CAPTCHA. A total of 210 participants of age between 19 and 64 participated during May and July 2010. The survey results validate the common belief that CAPTCHAs are still difficult for humans to solve. They also provide insights that can be applied to improve users' experience on interacting with CAPTCHA systems.

Mori began the work on finding objects in adversarial clutter for the purposes of breaking a visual captcha [60] They test object recognition techniques on Gimpy and

EZGimpy, examples of visual CAPTCHAs, which are now quite old and outdated (text based w/ distortions). At the time, EZ-Gimpy was used by Yahoo. These CAPTCHAs provide excellent test sets since the clutter they contain is adversarial; it is designed to confuse computer programs. They have developed efficient methods based on shape context matching that can identify the word in an EZGimpy image with a success rate of 92%, and the requisite 3 words in a Gimpy image 33% of the time. The problem of identifying words in such severe clutter provides valuable insight into the more general problem of object recognition in scenes. The methods that we present are instances of a framework designed to tackle this general problem. Lang et. al. propose a method of secure CAPTCHAs by impeding captcha breakers with visual decryption [46]. It is introduced as an extra layer of security on top of existing CAPTCHA implementations. It uses visual encryption to encrypt images, which are presented to clients like a CAPTCHA. Its purpose is to compress many sub-images into a small image format that humans can decode visually but is hard for automated systems due to decrypting overhead, and having to process more images to find the hidden image. This paper introduces visual encryption as a viable method to encrypt CAPTCHAs, and tests a prototype to measure how efficiently users can find them. It also measures whether this method could impede a real CAPTCHA breaker. Results show humans detect images within 16-33 seconds, and deciphering images is almost 100%. Estimates on CAPTCHA breaking benchmarks show automated systems would be slowed significantly, even assuming the image is found and decoded. As sub-images increase, humans can process the visually encrypted images faster than automated systems can.

Bursztein et al. bring bad news in 2014, with a study on the death of text based challenges. [7]. They note that over the last decade, it has become well-established that a CAPTCHAs ability to withstand automated solving lies in the difficulty of segmenting the image into individual characters. The standard approach to solving

CAPTCHAs automatically has been a sequential process wherein a segmentation algorithm splits the image into segments that contain individual characters, followed by a character recognition step that uses machine learning. While this approach has been effective against particular CAPTCHA schemes, its generality is limited by the segmentation step, which is hand-crafted to defeat the distortion at hand. No general algorithm is known for the character collapsing anti-segmentation technique used by most prominent real world CAPTCHA schemes. The team introduces a novel approach to solving CAPTCHAs in a single step that uses machine learning to attack the segmentation and the recognition problems simultaneously. Performing both operations jointly allows our algorithm to exploit information and context that is not available when they are done sequentially. At the same time, it removes the need for any hand-crafted component, making the approach generalizeable to new CAPTCHA schemes. Their experiments demonstrated that they were able to solve all the real world CAPTCHA schemes evaluated accurately enough to consider the scheme insecure in practice, including Yahoo (5.33%) and ReCaptcha (33.34%), without any adjustments to the algorithm or its parameters. The success against the Baidu (38.68%) and CNN (51.09%) schemes that use occluding lines as well as character collapsing leads us to believe that our approach is able to defeat occluding lines in an equally general manner. The effectiveness and universality of the results of the new approach suggests that combining segmentation and recognition is the next evolution of catpcha solving, and that it supersedes the sequential approach used in earlier works. More generally, our approach raises questions about how to develop sufficiently secure CAPTCHAs in the future.

CAPTCHAs themselves can now be used as a method of attack for data exfiltration as done in the work by Gelernter et. al. demonstrates [27]. They present the malicious CAPTCHA attack, allowing a rogue website to trick users into unknowingly disclosing their private information. The rogue site displays the private information to the user in obfuscated manner, as if it is a CAPTCHA challenge; the user is unaware that solving the CAPTCHA, results in disclosing private information. This circumvents the Same Origin Policy (SOP), whose goal is to prevent access by rogue sites to private information, by exploiting the fact that many websites allow display of private information (to the user), upon requests from any (even rogue) website. Information so disclosed includes name, phone number, email and physical addresses, search history, preferences, partial credit card numbers, and more. The vulnerability is common and the attack works for many popular sites, including nine out of the ten most popular websites. Note that the attack was evaluated using IRB-approved, ethical user experiments.

# Chapter 3

# CAPTCHA Theory & Design

## 3.1 Design Requirements and Challenges

We first discuss the CAPTCHA design considerations and the key requirements and challenges faced when developing new CAPTCHAs.

### 3.1.1 CAPTCHA Design Considerations

We are primarily interested in three major criteria by which a CAPTCHA's quality can be judged – usability, scalability, and robustness. Each one of these categories has effects on how the CAPTCHA manifests itself design-wise, and its ability to provide form security against various attack methods. It is worth noting that novel methods for creating CAPTCHAs are fairly straightforward to come up with (e.g., KittenAuth) [72], as just about any hard AI problem, logic puzzle, mind game or simple cognitive task can be used as the basis for a challenge. These are guaranteed to have a measure of success at stopping bot attacks already in use as long as they avoid or alter the methodology of the CAPTCHA that has failed. We present a number of these types of novel examples that we have created in Section 3.3.1. Unfortunately, very few of these challenges (if any) hold up to intensive scrutiny in the face of a tenacious and intelligent attacker with a toolbox of machine learning techniques and time on his side [64, 48, 30, 20, 19, 75, 2]. This fact only becomes more true as newer and more powerful algorithms and systems are devised over time [9, 58, 29]. Therefore, if there is ever to be any hope of securing online forms from a bot scourge, a scope analysis and a quantification of the security provided by a CAPTCHA must be done by the designer to ensure a realistic outcome and that a reasonable expectation of security can be maintained by the online service using the CAPTCHA.

To this end, the ReCAPTCHA service can be used to provide the benchmarks for usability, scalability and robustness for comparison to other CAPTCHAs, since it is maintained by Google and used across a variety of their own services in addition to third party sites. No other CAPTCHA can compare to its strengths and variability, and none of its weaknesses are so great that it fails to uphold form security to an acceptable degree. Its longevity, mutability (change over time), and adoption as the standard for online services / form control speaks to its success and the success of the methods it employs in general.

#### 3.1.1.1 Usability

Usability relates to all of the ways in which a user must interact with the CAPTCHA challenge and provide the solution. This idea can encompass numerous factors, some that are easy to quantify and measure, and some that are more qualitative/subjective. Quantifiable usability metrics include: average user time to solve, average number of challenges presented before correct response, server CPU time to generate challenge, etc. Qualitative usability metrics include: user reported ease of use on type of challenge presented, user/challenge method of interaction etc. Each of them plays an important role in the overall usability of a particular CAPTCHA implementation, and the interactions between each of the metrics and how they affect one another as well as the user must be considered. The critical consideration for usability revolves around the subjective question, “How easy is it for the user to solve this CAPTCHA challenge?” which can be measured quantitatively via a user study.

#### 3.1.1.2 Scalability

Scalability is perhaps the most important factor when creating a strong, secure challenge for an enterprise scale web service / form. Scalability refers to the ability to generate a large number of unique, one-off challenges and serve them to users at the rate deemed necessary to serve the userbase of the service provider. This is important since otherwise a database of the challenges can be maintained. While this definition might seem simple, it is perhaps the most challenging criterion to successfully achieve when designing a CAPTCHA. A question such as, “how easy is it for my CAPTCHA to generate a unique challenge?” falls within the realm of scalability. For example, it is relatively easy to see that a text based CAPTCHA can provide unique combinations of characters at a particular length that lends itself to an extremely large space complexity. Contrast this example with an image based CAPTCHA, which requires a tagged image database (expensive from a time, resource, and computational standpoint to create and maintain) to serve unique (or semi-unique, as at some point images must be re-used) challenges to the userbase.

#### 3.1.1.3 Robustness

Robustness is a measure of how well a particular type of challenge holds up against the various tools and techniques used to break CAPTCHAs. Is the challenge easy to break with OCR programs? Can a computer vision program accurately classify the images in a challenge? Would a competent scripter be challenged to write something clever that could foil the task posed by the challenge? What techniques or logic is the designer of the CAPTCHA using to hamper or foil these attacks? By logical extension, robustness also covers the degree to which challenges vary between one another within the same category of challenge. This category frequently has the most "turnover" of the three categories, meaning that what is considered robust today could easily be defeated by a new system that comes from new research.

### 3.1.2 Challenges in Designing a CAPTCHA

This section will help elucidate the thought process behind conception, evaluation and implementation of new and existing CAPTCHAs to secure an online service provider from bot attacks. We now discuss the various requirements and challenges in building and deploying CAPTCHAs to secure an online service provider from bot attacks.

#### 3.1.2.1 Risk Analysis Based CAPTCHA selection

The appropriate CAPTCHA to use depends on the environmental context and the degree of risk inherent in allowing access. The following questions can guide the CAPTCHA selection process:

1. What type of service is being provided? Can this service be considered "critical"?

2. What impact will a breach of the form security have on service operations? E.g. Service downtime, expanded operations costs, loss of goodwill from userbase etc.

3. What impact will a breach of the form security have on service's users?

4. What is the "cost" of supporting spam accounts? Cost can be measured in many ways, such as CPU time, bandwidth, account monitoring (automated and human), etc.

5. What is the level of tolerance built into the service? e.g. system operations-wise (quantifiable) for dealing with negative factors such as user annoyance at

CAPTCHA, at spam comments evading controls, loss of service, loss of users etc.

#### 3.1.2.2 Improved Machine Learning Tools

As time progresses forward, so does technology and algorithms. Computer vision in particular has made great strides in its general performance capabilities – an area where it could be considered lacking in the past, and is now providing new threats to CAPTCHAs. Thanks in part to the rise in GPU computing and cluster computing, compute power density is at an all time high. One recent example is the results from the 2014 Large Scale Visual Recognition Challenge which focuses on the task of image recognition in particular. The visual systems this year were tested in six categories based on their ability to detect objects, locate specific items in an object, and classify those images from a labeled dataset of 14 million images already tagged by humans. The average accuracy of entrants doubled from 22.5% to 43.9% while the error rate fell from 11.7% to 6.6% [53]. This level of success was achieved by most teams using an old algorithm – convolutional neural networks, which have been around for some time but were not practical until GPU computation was cheap and publicly available. While these systems are still no match for human analysis, it is easy to see why a CAPTCHA designer should become nervous about these systems. Indeed, the algorithm used in Google Streetview for recognizing the numbers on a house to determine the home address, which was based on deep convolutional neural networks, could equally be used to decipher the hardest category of ReCAPTCHA challenges they were serving, with 99.8% accuracy. Google's security blog [65] notes that they are moving away from text distortions as their primary method of security in their text based CAPTCHA, and instead have chosen to rely on performing advanced risk analysis – which they do not elaborate on for obvious reasons.

#### 3.1.2.3 An Economic Analysis

Motoyama et al.[61] published a seminal paper providing a thorough analysis of CAPTCHA breakers and their evolution into solving services that pay humans to solve CAPTCHAs. They make the argument that CAPTCHAs can increasingly be understood and evaluated in purely economic terms, that is, the market price of a solution compared to the monetizable value of the asset being protected. They examine the market-side of this question in depth, analyzing the behavior and dynamics of CAPTCHA-solving service providers, their price performance, and the underlying labor markets driving this economy. This analysis and the reality of solving/farming/mechanical turks as methods to beat CAPTCHAs at a near perfect rate (by the very definition of CAPTCHA) pose a grave threat to CAPTCHA use and the services they protect. Thus, Motoyama's paper successfully demonstrates that we must begin to look at the problem of online form security from an economic perspective, instead of a purely computer security perspective.

#### 3.1.2.4 The Insurmountable - Solving Services, Farming, & Mechanical Turks

CAPTCHA attacks can be abstracted to their logical extreme when an attacker simply pays a mechanical turk (a human via a web service) for the solution to a CAPTCHA. As long as the attackers' costs remain lower than his profits, he will use that method and by the definition of a CAPTCHA (correct solution to challenge determines human or bot), this "attack" will succeed, simply because a human is providing the solution. We can even humor the scenario where the attacker is content to operate at the break-even point (costs of attack are equal to profits), just to spite the service provider and antagonize legitimate users – a losing proposition for the defender. After spending time researching the area of CAPTCHAs, it is easy to fall into a fatalistic and nihilistic mindset with regards to public online service defense, especially when enterprise usability and scalability are critical factors. Given the advances in computational ability to provide near-human cognition [19, 58, 29, 61, 74], all CAPTCHAs can and will eventually be defeated by attackers and form security mechanisms will be forced to evolve into something else yet to be developed, or replaced entirely. However, until that day arrives, we must be content with utilizing a risk based framework combined with an economic analysis to select the strongest possible design criterion for CAPTCHAs and accept that pay to play solutions cannot be realistically stopped, short of extreme measures on behalf of the service provider.

## 3.2 Evaluation of CAPTCHA Styles

There are three major styles of CAPTCHA - each with its own unique angle for presentation to the user and its accompanying underlying methodology. These can then be subdivided based on various methods of implementation and/or various methods of testing for user humanity. Note that these methods correspond with 2 of the 5 human senses (sight and sound) that are easy to interact with via a computer. The omission of smell and taste being somewhat obvious, however touch is quickly becoming a new area of exploration as the technology to support this has recently moved into mainstream computing.

- Text

  This is the defacto standard style of CAPTCHA because it is the easiest to design, implement and use. It consists of a string of characters (letters, numbers, and sometimes special characters) that a user must type in to prove they are human. These are the most common types of CAPTCHA currently in use, due in part to their scalability (easy to generate) and robustness (uniqueness, space complexity).

- Image

These CAPTCHAs utilize images as a method to provide security. More specifically, they ask users to perform cognitive tasks that involve the images, such as image recognition (e.g. what is being shown in the image) or a categorization task (e.g. select all pictures of cats) based on viewing and comprehending what the images are depicting. These are the second most common type of CAPTCHA, although they often score high in usability, particular implementations provide shortcomings in the scalability and robustness category, and are weak against certain types of attacks. We will demonstrate some of the shortcomings of image CAPTCHAs in these two categories in our design overview.

- Audio

  These CAPTCHAs are usually used in tandem with another CAPTCHA (usually text based) to provide accessibility to blind users. The audio CAPTCHA speaks the characters out loud so that blind users can understand. They usually are susceptible to speech to text attacks and can pose some usability problems - namely the protection used (other garbled noise to disguise the characters) to make them more secure causes audio CAPTCHAs to become nearly impossible to decipher for the end user and bots alike. Also, not all computers have speakers or a port accessible for headphones - thus there is no guarantee of sound at any particular computer terminal.

- Multi-Modal

  This method takes two or more existing, for example image and text based methods, and combines them to provide enhanced usability [12, 4].

### 3.2.1 Alternative CAPTCHAs – An Overview of Existing Works

This section provides a quick overview of existing alternative methods of CAPTCHA that are designed to be more robust methods beyond traditional simple text, image, or audio based CAPTCHAs. Many of these styles incorporate nuanced information contained in text or images into the challenge question to provide stronger security.

- *HIP*

  Human Interactive Proofs - To be effective, a HIP must be difficult enough to discourage script attacks by raising the computation and/or development cost of breaking the HIP to an unprofitable level. The purpose of this study was to find the visual distortions that are most effective at foiling computer attacks without hindering humans. This was achieved by building segmentation-based HIPs that are extremely difficult and expensive for computers to solve, while remaining relatively easy for humans.[11]

- *GOTCHA*

  Generating panOptic Turing Tests to Tell Computers and Humans Apart  A GOTCHA is a randomized puzzle generation protocol, which involves interaction between a computer and a human. Informally, a GOTCHA should satisfy two key properties: (1) The puzzles are easy for the human to solve. (2) The puzzles are hard for a computer to solve even if it has the random bits used by the computer to generate the final puzzle — unlike a CAPTCHA. GOTCHAs are generated on the fly by a series of mathematical equations and require the users to describe the resulting image with a phrase. This is primarily used as a password replacement system.[6]

- *POSH*

  Puzzle Only Solveable by Humans  this method has three primary criteria: it can be generated by a computer, it can be consistently answered by a human, and a human answer cannot be efficiently predicted by a computer. The designer suggests that a POSH does not even have to be verifiable by a computer at all. Everything that is a CAPTCHA is by definition also a POSH, but not vice versa. For example, "What is your favorite food?" is not a valid CAPTCHA (there is no correct answer) but is a valid POSH. The paper explores this space further by investigating what features make for a desirable POSH, what constraints affect the POSHes that can be reasonably created, and which POSHes are actually fun to solve. [15]

- *Whats UP*

  This is an image based CAPTCHA in which the challenge task centers on identifying an image's upright orientation. Given a large repository of images, such as those from a web search result, What's up uses a suite of automated orientation detectors to prune those images that can be automatically set upright easily. The application of a social feedback mechanism to verify that the remaining images have a human-recognizable upright orientation is used to identify the rest of the challenge images. The main advantages of this CAPTCHA technique over the traditional text recognition techniques are that it is language-independent, does not require text-entry (e.g. for a mobile device), and employs another domain for CAPTCHA generation beyond character obfuscation.[32]

- *Image Flip*

  This CAPTCHA asks a user to correctly choose the orientation of a number of images by clicking on them presented in a grid that have had distortions applied to them to. In the proposed technique a composite CAPTCHA image of a reasonable dimension and resolution is shown to the user. The user has to identify positions of all embedded images that appear as normal with no ip applied to them from the shown composite image. The user needs to click on every non-ipped embedded image to prove human interaction.[52]

- *CORTCHA*

  Context-based Object Recognition to Tell Computers and Humans Apart - using images gathered from the internet, the designers note that although an object that is segmented by a computer might be poor cognitively, but if the object is surrounded by its original context in the image, then the object is readily recognizable by humans. By exploiting the context, objects segmented by computer can be used in an Image Recognition CAPTCHA (IRC). The use of context solves the dilemma, and an IRC can be designed without labeling any image. [78]

- *SKETCHA*

  This CAPTCHA uses line drawings of 3D models rotated to a randomized point of view. The goal of the user is to rotate each image until it is upright, choosing among four orientations by clicking on the image. Each line drawing was automatically rendered from a 3D model using a randomized point of view, providing for many possible images from each model. Solving this challenge generally requires understanding of the semantic content of the image, which is believed to be difficult for automatic algorithms. The authors also cover a process called covert filtering used by the CAPTCHA whereby the image database can be continually refreshed with drawings that are known to have a high success rate for humans, by inserting randomly into the CAPTCHA new images to be evaluated.[63]

- *IMAGINATION*

  IMAge Generation for Internet AuthenticaTION - produces controlled distortions on randomly chosen images and presents them to the user in the form of a mosaic image. The authors recommend the use of a two step verification process. In the first step, the user clicks near the geometric center of any picture in the mosaic, which is composed of a number of images of various sizes merged into a single large image that have had distortions and alterations applied to the images. In the second step, the user is asked to identify a distorted image by selection from a list. This two-round click-and-annotate process makes the CAPTCHA user friendly and very effective.[18]

- *Interactive Games*

  One example of an interactive game CAPTCHA is FunCAPTCHA. It asks the users to perform two "fun" tasks to prove they are human. These tasks usually take the form of small games, such as selecting the picture of a woman from 9 pictures and drag it to the middle of the CAPTCHA, or rotate the image until it is facing up. [35]

- *Video CAPTCHA*

  There are a number of video CAPTCHAs available in the market. NuCAPTCHA is one of the more popular implementations where a string of characters is presented to the user within the video. The intention is that encapsulating the challenge within the video makes it more difficult to access (timing, location of challenge string etc.) than traditional text based challenges. The other style (SolveMedia) shows the user an advertisement and asks them to type something related to the video advertisement that has been shown to them. [38]

- *Image CAPTCHA - Real World Distances*

  This image CAPTCHA scheme is based on a human's understanding of real world objects and their relative distances. Solving this CAPTCHA involves answering questions about the relative distances of objects present in an image. As an object moves farther away from the point of perception, its size decreases.

But equipped with only the information on size, a decision cannot be made on its position from the point of perception. The real world knowledge of the object dimensions is essential to distinguish between them since both a large and small object near the perception point will appear similarly sized and cannot be differentiated. This ability of humans, aided with experience, is a unique ability and would thus serve to distinguish them from computers. Examples of the challenge questions are - “who stands behind whom”, “which is nearer” or “which is larger in real life”.[1]

- *ASIRRA*

  Asirra (Animal Species Image Recognition for Restricting Access) is a HIP that works by asking users to identify photographs of cats and dogs. At the time of its creation (2007), this was a challenging task for computer vision and machine learning to defeat. Asirra surmounts the image-generation problem in a novel way: by forming a partnership with Petfinder.com, the worlds largest web site devoted to finding homes for homeless pets. Asirra generates challenges by displaying 12 images from a database of over three million photographs that have been manually classified as cats or dogs. Nearly 10,000 more are added every day by volunteers at animal shelters throughout the United States and Canada. The size and accuracy of this database is fundamental to the security provided by Asirra.[21]

- *Jigsaw Puzzle*

  An image is divided into an n (n= 3, 4 or 5, depending on security level) pieces to construct the jigsaw puzzle CAPTCHA. Only two of the pieces are misplaced from their original positions. Users are required to find the two pieces and swap them. Considering the previous works which are devoted to solving jigsaw puzzles using edge matching techniques, the edges of all pieces are processed with a glitch treatment to prevent automatic solving.[25]

- *Scene Tagging*

  This CAPTCHA tests the ability to recognize a relationship between multiple objects in an image that is automatically generated via composition of a background image with multiple irregularly shaped object images, resulting in a large space of possible images and questions without requiring a large object database. This composition process is accompanied by a carefully designed sequence of systematic image distortions that makes it difficult for automated attacks to locate/identify objects present. An experimental study using several widely-used object recognition algorithms (PWD-based template matching, SIFT, SURF) shows that the system is resistant to these attacks with a 2% attack success rate, while a user study shows that the task required can be performed by average users with a 97% success rate.[54]

- *Multiple SEIMCHA*

  The Multiple SEIMCHA system warps images by using geometric transformations and a 2D view of the warped image is shown to the user. Users click on the upright orientation of warped image as a semantic to solve the challenge. This CAPTCHA also experiments with the idea of "almost right" in that it evaluates the difficulty of the challenges solved over time and compares them to a database which stores the historic response rate. The more difficult the challenge is to solve over time, the more of a break the user gets when solving the challenge.[55]

- *Categorizing CAPTCHA*

  In this method, a number of objects are chosen randomly and the pictures of these objects are searched in the Internet and downloaded. The pictures are then shown to the user and the user is asked to mark the objects which belong to a specific category. If the user marks the right objects, it can be assumed that the user is a human being and not a computer program.[66]

- *THINK*

  This CAPTCHA uses a real time image which will portray some action or show some object that the user is expected to identify the object and type the answer. No choices are given to the user thereby eliminating the option of identifying the answer by probability. Questions are framed related to the picture by the human and as well the expected answers ie., the keywords. The work of the computer is to randomly throw these images with questions to the user and compare the answers given by the users with the key words and conclude whether the user is a human or a bot based on the answer. [68]

- *Dyanmic Image Based CAPTCHA*

  The authors of this CAPTCHA propose an image CAPTCHA that can be divided into three separate layers - the layers are 1) Image access layer 2) Image processing layer and 3) Presentation layer. These layers are designed to provide security for a typical IRC challenge. The authors of this challenge provide an in-depth overview of how they created a new challenge consisting of 32 filters that they can apply to images in a specific order to ensure human usability while being able to provide protection from bots.[70]

- *SEMAGE*

  SEMAGE (SEmantically MAtching imaGEs), a new image-based CAPTCHA that capitalizes on the human ability to define and comprehend image content and to establish semantic relationships between them. A SEMAGE challenge asks a user to select semantically related images from a given image set. SEMAGE has a two-factor design where in order to pass a challenge the user is required to figure out the content of each image and then understand and identify a semantic relationship between a subset of them.[71]

- *3D Object CAPTCHA*

  This CAPTCHA is a novel 3D object based CAPTCHA scheme that projects the CAPTCHA image over a 3D object. The CAPTCHA is a traditional distorted text based challenge. The challenge question first asks a user to solve the 3D object rotation problem, similar to Sketcha. Then, a user has to solve the 3D text CAPTCHA.[73]

### 3.2.2 New HCI, Sensors & Biometrics Opportunities for Bot Detection

- *HCI – Beyond KB & Mouse (Touch Screens/Pen Input/Haptics)*

  As computing devices change to suit the needs of the user, we are witnessing a move towards mobility  laptops, tablets, smartphones etc. Many of these devices have new methods of input beyond traditional keyboards and mice. These devices feature input like touch sensitive screens and stylus pen input devices. These provide new opportunities to explore interactive CAPTCHAs that would have been challenging or impossible with keyboards and mice. For example, a drawing or tracing CAPTCHA that asks a user to draw random shapes around random image objects to prove they are human.

- *Sensor based CAPTCHAs*

  Smartphones and tablets have a number of different sensors in them, such as accelerometers, gyroscopes, gps radios, cameras, compass, etc. These can be used by clever designers to get a user to perform an action that would be difficult for a computer or have a series of events transpire where the measurements are taken from each of these devices as a human performs an action in real life and if they match the challenge action, the user is verified. Special "human impossible" actions can be sprinkled in to catch bots if they attempt to feed information to the sensors to pass the challenges.

- *Biometrics*

  Biometrics are not CAPTCHAs but they have an analogue in that they both require a human/human input or something that is uniquely human and use it as a security method. Biometric based CAPTCHAs can become a new field as biometrics begin to disseminate amongst the public. A prime example is the new Apple iPhone and iPads having fingerprint sensors that provide verification for Apple's services or serving as a password replacement. While biometrics are not foolproof, as they improve in accuracy they will provide reasonable, easy to use security.

### 3.2.3 Future CAPTCHAs - Some Proposals

It is a worthwhile exercise to speculate about the future direction of CAPTCHAs, as they become more important to the smooth, everyday operation of public websites. As human-computer interfaces and cybernetic devices advance, we must postulate on their ramifications to online form security - particularly methods that attempt to discern human from bot. A natural area to begin a search is with other human senses that are currently not being used in CAPTCHA challenges today. Many of these ideas we present here are far-fetched at best - while they are not impossible to implement, they almost assuredly will fail to meet our three criteria in some capacity. Many of these ideas blur the line into biometric security and can leverage those technologies for online form security uses.

- *Smell & Taste CAPTCHAs:* These CAPTCHAs could be implemented via a challenge where a small machine containing a number of base chemicals creates a scent/flavor that a human must smell/taste (think smell-o-vision). For example, a smell challenge could consist of a series of 4 or 5 smells produced by the machine in a random order - at the conclusion of the smell session, an image can be displayed. The human must select the smell that corresponds with the image. For a taste CAPTCHA, a flavor can be produced via mist by the machine for human consumption - at the conclusion of the taste session, a series of images can be displayed. The human must select the flavor that corresponds to the correct image. The fun in this machine is that consecutive incorrect responses to challenges or multiple simultaneous solutions from the same IP could be met with increasingly foul tastes, discouraging mechanical turks or farming. We can imagine an attacker with an "artificial nose/tongue" could detect the ppm of the various chemicals and map them to various images.

- *Eye Movement CAPTCHA:* This challenge requires a webcam and specialized tracking software that analyzes human eye movements. There is a strong possibility this could run on a smartphone with a front facing camera as well. A randomized pattern could be displayed on screen that the users eyes must follow. If the program detects the correct eye movements, the user is verified. A camera flash could be used to attempt to force the user's pupils to dilate, which could then be measured by the software accordingly. The random nature of the points and the flashes of light prevents an attacker from using a pre-recorded video of eyes/eye movements to crack the challenge. However, the usability requirements mean that some margin of error in the measurement of eye movements is required that an attacker will probably exploit (hardware viability notwithstanding).

- *Brain Wave CAPTCHA:* This challenge requires the user to wear a specialized piece of equipment on their head or face that can measure their brainwaves. Imagine a device similar to Google glass but with electrodes on the temples of the person and pinhole cameras on the frames facing inward towards the users eyes. A series of images can be shown to the users that have been shown to stimulate various parts of the brain and produce alpha, beta, and gamma waves. These can be randomized by the CAPTCHA so that each time the user needs to verify, a different unique sequence is provided to them. In the time since we came up with this idea (Late 2012), the U.S. Army has successfully developed and implemented this technology in a prototype capacity.

## 3.3 Methodology, Design, and Analysis

In this section, we have provided a few examples of CAPTCHAs we have thought about implementing to test out as effective methods of performing reverse Turing tests. We structure each CAPTCHA by type - providing a brief overview of the concept and how to solve the challenge, a discussion of its functionality focused on its usability and its scalability, and its security strengths and weaknesses. Note that while these CAPTCHAs can work they can only do so on a very small scale and most would fail in the scalability category for large public facing web services. Our intentions of providing these is to demonstrate why the text based (especially ReCAPTCHA) CAPTCHA is unlikely to be replaced anytime soon – at least for enterprise scale, public websites, despite recently being shown to be vulnerable to deep learning neural networks.

When "Warping" is referred to in the style analysis tables, it refers to all of the traditional obfuscations that have been used with text based CAPTCHAs in the past. Things such as overlapping characters, distortion of the characters via image filters, additional noise, color blending etc. can all be used to secure text from OCR and CV attacks, at the expense of usability. While the effectiveness of these techniques at reducing attacks is coming under increased scrutiny as tools evolve, it is still provides an impedance and can provide benefits when combined with other obfuscations.

A brief note about the “Cambridge effect” that is referenced throughout this section - this refers to the unique ability for humans, English speakers in particular, to read words that are misspelled as long as the first and last letters of the word remain in their correct location within the word. This essentially provides a method to “scramble” words that are 4 or more characters in length, and provide an additional layer of obfuscation for challenge keywords, phrases or sentences. The “Cambridge effect” is actually an old internet hoax, as no research at Cambridge University has been conducted for this topic – nevertheless it serves a useful purpose in our CAPTCHA design. The following paragraph in italics is the original example of “the Cambridge effect” in action:

*Aoccdrnig to rscheearch at Cmabrigde uinervtisy, it deosn't mttaer waht oredr the ltteers in a wrod are, the olny iprmoetnt tihng is taht the frist and lsat ltteres are at the rghit pclae. The rset can be a tatol mses and you can sitll raed it wouthit a porbelm. Tihs is bcuseae we do not raed ervey lteter by it slef but the wrod as a wlohe.*

The following section demonstrates how we apply the CAPTCHA design criteria and risk analysis framework to various challenge ideas to perform an analysis of CAPTCHA security. The goal is to demonstrate the process of evaluating ideas from conception to application of design criteria followed by a risk analysis. This is done for a number of ideas our research group came up with. Note that security analysis portion essentially serves as the criteria evaluation for “robustness” as a category of design.

### 3.3.1 Application of Design Criteria to Challenge Ideas

#### 3.3.1.1 Word Change CAPTCHA

This CAPTCHA uses 4 words, where each word differs from another by 1 letter, which alters the meaning of the word. The words are represented by an image. The goal is for the user to be able to determine the sequence of the words and come up with the correct final word.

Usability: Easy to use for those who speak English and can understand the spelling of the words depicted in the image, so as to understand the change / shift.

Scalability: Somewhat difficult to scale as it requires real words and chaining techniques be used.

Strengths: Difficult for OCR to crack. Image recognition will require large database.

Weaknesses: Small scale means vulnerability to database attacks, as well as lexicographical attacks. Farming attacks are also a problem.

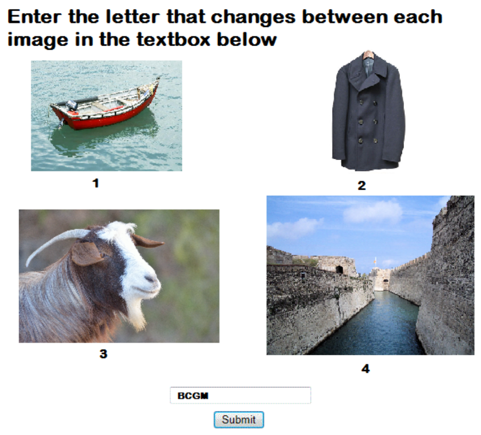

Enter the letter that changes between each image in the textbox below
1
2
3
4
BCGM
Submit

Figure 3.1: Example of word change challenge IMAGE: ©CIMIC Lab - Rutgers University

### 3.3.1.2 Storyboarding CAPTCHA

In this CAPTCHA the user is presented with a randomized set of panels from a short comic. These panels have a definite order and must be placed in this order. The user needs to drag the panels into the correct order. The CAPTCHA can additionally benefit from Cambridge effect for increased security and complexity.

Usability: Easy for a user to understand as the story in the comic will not make sense unless the panels are put in the correct order. Dragging the panels requires that javascript be used / enabled.

Scalability: Difficult to scale, as comic strips must be created, segmented, and tagged to ensure the generation algorithm understands the correct order. Potential for automation exists via mining old published strips, but is difficult to achieve without prescreening for meeting requirements. Licensing can also be an issue.
Strengths: Extremely difficult for adversary to comprehend story without advanced NLP and DL reasoning capabilities.
Weaknesses: Small scale means vulnerability to database attacks, also vulnerable to farming attacks.

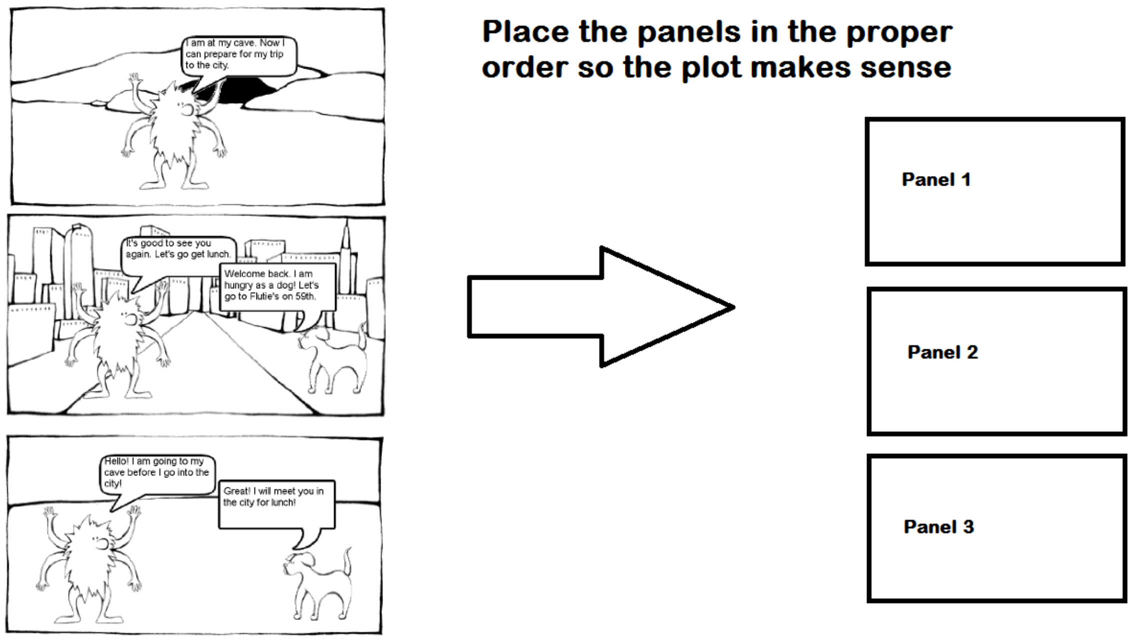

I am at my cave. Now I can prepare for my trip to the city.
It's good to see you again. Let's go get lunch.
Welcome back. I am hungry as a dog! Let's go to Flutie's on 59th.
Hello! I am going to my cave before I go into the city!
Great! I will meet you in the city for lunch!
Place the panels in the proper order so the plot makes sense
Panel 1
Panel 2
Panel 3

Figure 3.2: Example of storyboard challenge IMAGE: ©ILA/NCTE 2016

#### 3.3.1.3 Consequence & Conclusion Image CAPTCHA

In this CAPTCHA two images are presented that represent an idea where the final outcome is the result of the two images interacting. Ex: New York City + Hurricane Irene = Flooded city. The user needs to choose the image that represents the correct outcome of the interaction.
Usability: The user must make a selection based on their reasoning of what the images

are trying to communicate. The CAPTCHA is simple in that it requires selecting an image.

Scalability: The CAPTCHA is difficult to scale, as it requires a large, tagged, relational database of consequence and conclusion images.

Strengths: Extremely difficult for adversary to comprehend story without advanced NLP and DL reasoning capabilities. Nothing for OCR to pick up.

Weaknesses: Small scale means vulnerability to database attacks, also vulnerable to farming attacks.

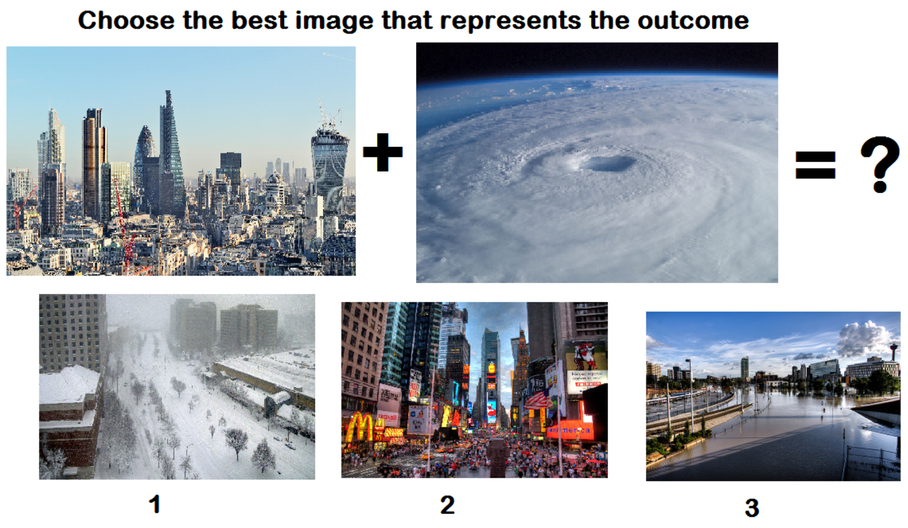


Figure 3.3: Example of consequence and conclusion challenge IMAGE: ©CIMIC Lab - Rutgers University

#### 3.3.1.4 Pattern Completion CAPTCHA

This CAPTCHA is designed to act as a method to counteract OCR and Image recognition techniques. The CAPTCHA requires a user to order a random series of images, words, or numbers in ascending or descending order (also randomized). The user must correctly order the challenges based on the solution dictated by the CAPTCHA. The CAPTCHA may additionally benefit from Cambridge effect for increased security and complexity.

Usability: The CAPTCHA is difficult from a usability standpoint, as it requires users to reason through a number of qualifiers (i.e. ascending/descending, ordering from greatest to least etc.), which might be too complex for some users.

Scalability: Easy to scale as qualifiers can be randomized and the increments generated at random as well.

Strengths: Multiple layers of security from randomized ascending/descending and greatest/least ordering.

Weaknesses: Vulnerable to image recognition attacks and OCR attacks, as the indicators can be deciphered with the proper tools and techniques.

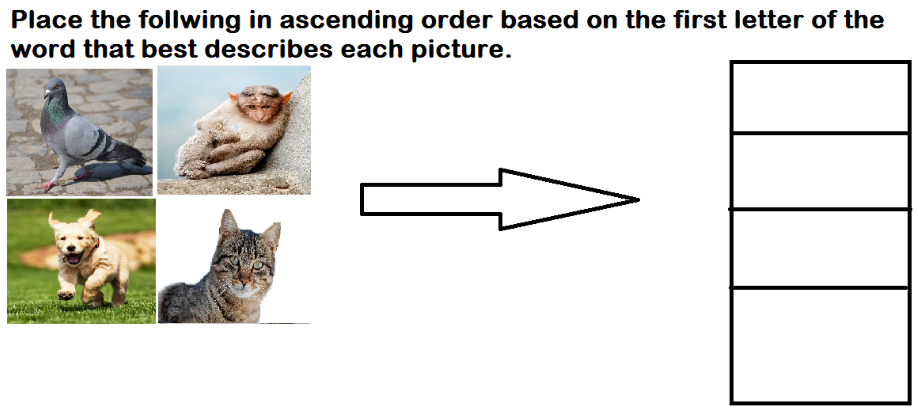

Place the follwing in ascending order based on the first letter of the word that best describes each picture.

Figure 3.4: Example of pattern completion challenge IMAGE: ©CIMIC Lab - Rutgers University

### 3.3.1.5 Pictionary CAPTCHA

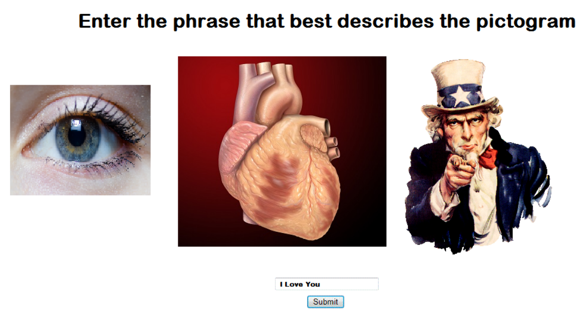
In this CAPTCHA the user is presented with a series of images that represent a short English phrase. Ex: I love you communicated by a picture of a human eye, a heart, and a finger pointing toward the user. The user must correctly decipher the phrase and select or type the matching answer.

Usability: This CAPTCHA is somewhat difficult to use, in that the images used in Pictionary must communicate the idea to the user. There is a high probability of misinterpretation or miscommunication of the idea required to solve the challenge.

Scalability: Easy to scale, as the ideas represented by the images form a sort of symbolic language that can be mixed and matched accordingly.

Strengths: Strong against OCR as there are no words to pick up.

Weaknesses: Vulnerable to a combination image recognition and dictionary attack, as someone could create an equivalency chart for pictures to ideas. Also vulnerable to farming attacks.


Enter the phrase that best describes the pictogram

I Love You

Figure 3.5: Example of pictionary challenge IMAGE: ©CIMIC Lab - Rutgers University

#### 3.3.1.6 Jigsaw Puzzle CAPTCHA

In this CAPTCHA an image is cut into 9 or 12 equal squares and scrambled. The user is asked to place the pieces into the framework to correctly reassemble the image within the allotted time. The user must reconstruct the image inside of the frame by dragging the pieces to their correct places.

Usability: This CAPTCHA has moderate usability as the person must be able to put the puzzle together in a reasonable amount of time, without making it too easy for an image processing algorithm to do the same.

Scalability: Easy to scale as it only requires an image that meets a certain set of criterion, aligned with the purpose of defeating image / edge detection algorithms.

Strengths: Strong against OCR, as there are no words to pick up. Depending on the level of the implementation sophistication, provides some resistance against edge detection / segmentation attacks.

Weaknesses: Vulnerable to tagged image database, image recognition, and reverse image search.

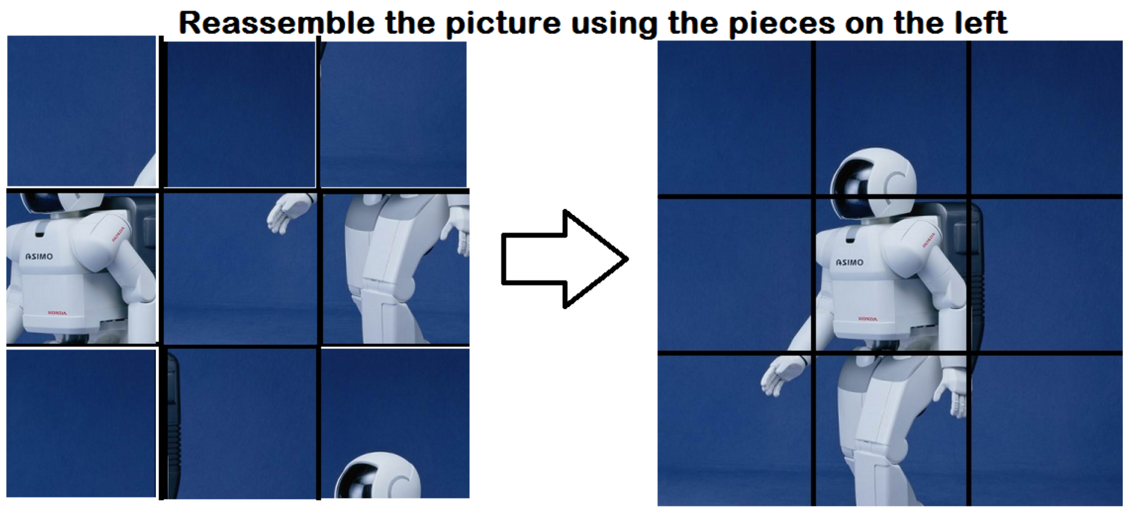


Figure 3.6: Example of jigsaw puzzle challenge IMAGE: ASIMO ©Honda Inc.

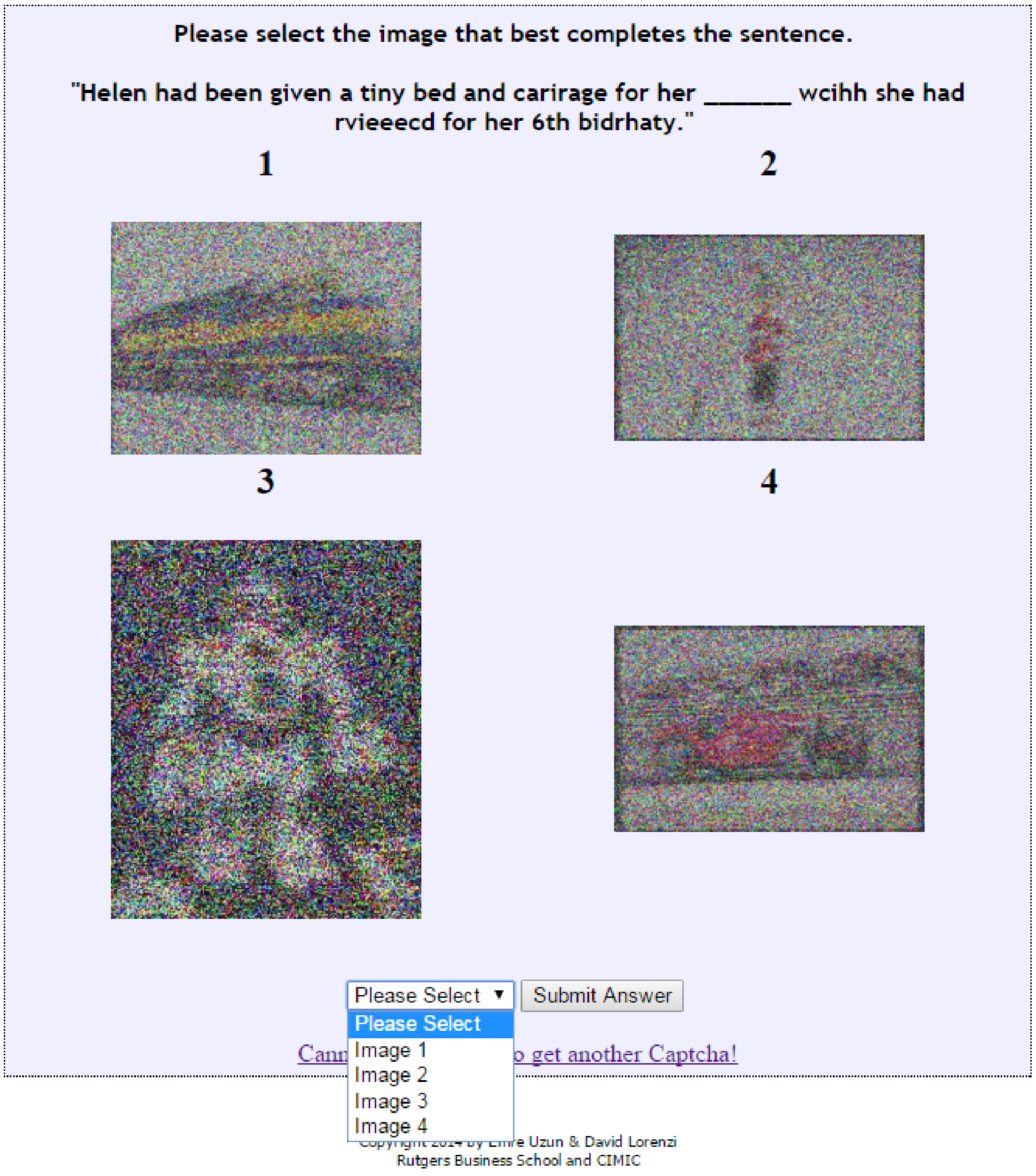
#### 3.3.1.7 Cambridge Study CAPTCHA

This CAPTCHA is based on the above mentioned (apocryphal) Cambridge study which notes that as long as the first and last letters of a word are in the correct place, the rest of the word can be scrambled and is still readable to the average English reader. Ex: Cantaloupe to Cnatalupoe. A randomized number of sentences in English are collected and scrambled based on this idea. A word is then removed from the sentences and the user must select the correct word that completes the sentence from the list. Thus, to solve this CAPTCHA the user must correctly decipher the word and pick the appropriate word to fill in the blank in the sentence. An alternative to this is to select and scramble a word and then to present the user with a list of words, where 3 are similar to the scrambled word (synonyms) and 1 is different (antonym), now requiring the user to select the three synonyms without selecting the antonym.
Usability: As the study holds true, it should be easy for speakers of English to understand the words, even when scrambled in this fashion.

Scalability: Easy to scale, as any word that is 4 or more letters long in the English language can be scrambled in this fashion and sentences can be pulled from online libraries, catalogs, etc. Note that the alternative version looks for comprehension of the word, as opposed to just an unscrambling / word match and can pull words from dictionary and thesaurus.

Strengths: Strong against image recognition, as it is a word based CAPTCHA. OCR weakness can be mitigated through warping.

Weaknesses: Vulnerable to lexicographical attacks (i.e. scrabble algorithm looking for dictionary hits) and OCR. The alternative version is potentially vulnerable to dictionary attacks and reverse thesaurus lookup (i.e. find the antonym)



Please select the image that best completes the sentence.

"Helen had been given a tiny bed and carirage for her _______ wcihh she had rvieeecd for her 6th bidrhaty."

1
2
3
4
Please Select
Submit Answer
Please Select
Image 1
Image 2
Image 3
Image 4
Cann o get another Captcha!
Copyright 2014 by Emre Uzun & David Lorenzi
Rutgers Business School and CIMIC

Figure 3.7: Example of Cambridge study used in image CAPTCHA challenge with SIGNAC applied IMAGE: ©CIMIC Lab - Rutgers University

#### 3.3.1.8 Compound Words CAPTCHA

This CAPTCHA uses compound words to generate two images that represent the requisite two words contained in the images. It is flexible in that you can show the pictures and have the users select the words or show the words and have the users select the pictures. The goal is to get the idea communicated by the word/images and match the idea to the answer. Alternatively, the CAPTCHA can also provide a compound word where the user must select an image associated with another compound word to illustrate the idea of the compound word. Ex: Sunflower represented as two images featuring sunlight and flowerpot as the main focus. In this case the user must select the nuanced compound words that communicate the idea in the originally displayed word.

<u>Usability:</u> The CAPTCHA is easy to use in that it requires the user to select images or words that match with the idea communicated in the two or more images that make up the compound word. The alternative version has moderate usability as the ideas communicated can be too nuanced for the user to easily comprehend. Requires users to extract a word represented by an image. The user then selects the correct images with the constituent parts to make the original compound word.

<u>Scalability:</u> Moderate difficulty in scaling, as there are only a finite amount of compound words and tagged images are required for the generation of CAPTCHA.

<u>Strengths:</u> Moderate strength against image recognition attacks, as the combo image word requires intelligent link Strong against OCR attacks if using the images. The alternative version has moderate strength (superior than the basic version) against image recognition attacks, as combo image requires an “intelligent link”. Strong against OCR attacks if using the images.

<u>Weaknesses:</u> Tagged image database attacks, small list of compound words makes dictionary attack strong.

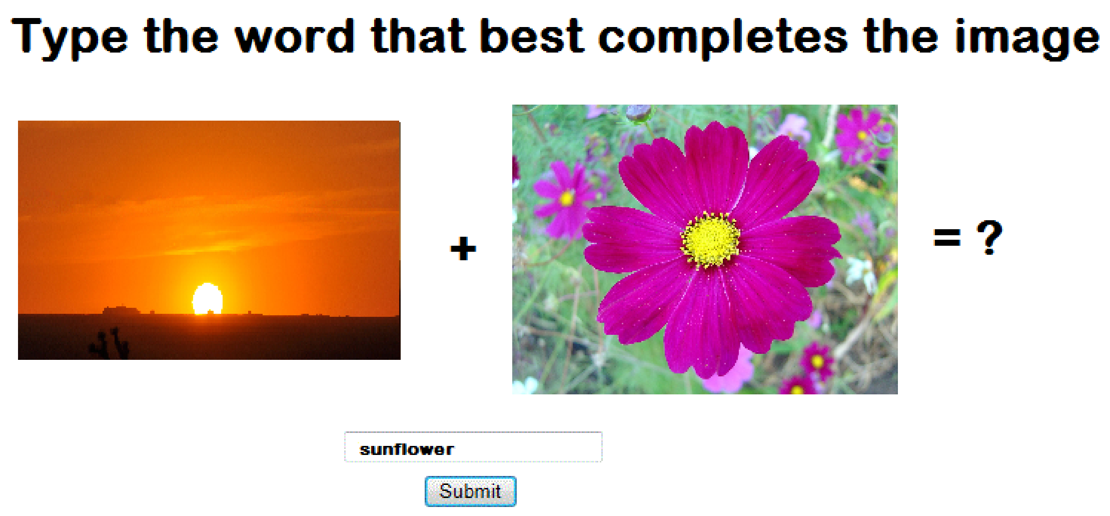


Figure 3.8: Example of compound words challenge IMAGE: ©CIMIC Lab - Rutgers University

### 3.3.2 Definitions for Trade-Offs in Design

In this section, we will briefly discuss the meaning and qualifications of the criteria low, medium, and high for each of the three design categories accordingly.

#### 3.3.2.1 Usability

Low: Users have a difficult time solving challenges. Challenges frequently require significant amounts of time to solve (e.g. greater than 45 seconds) and users often fail to solve the challenge correctly greater than 50% of the time. Security methods used in the challenge may be too extreme (e.g. too much noise/distortion) that makes comprehension for humans difficult. Challenge comprehension is difficult for average target user in native language, language barriers, and specialized symbol packages/fonts required. Implementation requires specialized browser plugins/extensions/software libraries that may not be widely available or have a large installed userbase and/or could provide security vulnerabilities to users and implementers.

Medium: Users can solve the challenge most of the time (e.g. greater than 75% of the time). Challenges require a moderate amount of time to solve (30 seconds or less).

Security methods used in the challenge may occasionally impact comprehension (e.g. too much noise/distortion in less than 25% of challenges) that makes comprehension for humans difficult. Challenge comprehension is easy for average target user in native language, does not require additional fonts/special character packages. Little to no language barrier (beyond directions). Implementation does not require specialized browser plugins/extensions/software libraries that are uncommon (i.e. common ones like javascript are OK).

High: Users have an easy time of solving the challenge quickly (10 seconds or less) and have a success rate of 90%+ for solving the challenges. Security methods do not impact challenge comprehension. Comprehension of challenge transcends languages and cultures and can be easily solved by all humans with a simple explanation in native language. Implementation does not require specialized browser plugins/extensions/software libraries that are uncommon (i.e. common ones like javascript are OK).

#### 3.3.2.2 Scalability

Low: Challenges must be crafted in part by a human activity (e.g. tagging images, drawing panels, labeling scenes etc.) that cannot be easily farmed out to a mechanical turk service. Ability to create a unique challenges is difficult through automation or number of unique challenges able to be presented to users is very small. Automation of challenge creation is very difficult.

Medium: Challenges must be crafted in part by a human activity (e.g. tagging images, drawing panels, labeling scenes etc.), but the core task can be scaled to a mechanical turk service. Ability to create a unique challenge is directly related to the ability to farm the core task to the mechanical turk service (e.g. uniqueness is based on variation in core task). A break even is achieved when the cost of generating a unique challenge is equal to the cost to pay a solving service to crack it. Degree of automation of challenge creation is directly dependent upon mechanical turk service.

High: Challenge generation is fully automated and does not require any human input. Large number of unique challenges can be created on-demand, and in parallel.

#### 3.3.2.3 Robustness

Low: Challenges are easily defeated using existing computer vision and optical character recognition tools. Attacker can successfully solve more than 10% of the challenges using fully automated tools in under 30 seconds. Core challenge does not require any higher level reasoning beyond identification task. Challenges are not unique and repeat frequently.

Medium: Challenges can potentially be defeated by existing computer vision and optical character recognition tools. Attacker must perform more than one additional step to defeat security methods in challenge and automation of attacks requires advanced knowledge of cv algorithms and creation of custom tools. Challenge questions repeat infrequently and are varied in their composition.

High: Challenges are immune to most/all known computer vision and optical character recognition attacks. Attacker cannot successfully solve 1% or more in an automated fashion in under 30 seconds. Large variety in challenges with zero repeated challenges or a very large challenge database such that repeats happen very rarely.

#### 3.3.2.4 Design Criteria Comparison Matrix

Table 3.1 gives a comparison of all of the above CAPTCHAs in terms of the design criteria of usability, scalability, and robustness.

| CAPTCHA Type | Usability | Scalability | Robustness |
|---|---|---|---|
| Word Change | Medium | Medium | Medium |
| Storyboarding | High | Low | High |
| Consequence and Conclusion | High | Low | High |
| Pattern Completion | Medium | Medium | Medium |
| Pictionary | Medium | Low | Medium |
| Jigsaw Puzzle | Medium | Medium | High |
| Cambridge Study | Medium | High | Medium |
| Compound Words | High | Low | Medium |

Table 3.1: Tradeoffs between each design criterion

# Chapter 4

# Attacks on IRCs

In this section, we will cover the two attacks we have crafted that can be used against particular implementations of image based CAPTCHAs. As with any security arms race, these attacks either become more powerful or weaken over time as tools and security methods improve.

## 4.1 Methodology & Design for HTM Based Attack Systems

This section provides an overview of the tools and techniques used to create the framework for a generalized attack methodology using hierarchical temporal memory(HTMs) models for use against IRCs. The specifics of each attack will be detailed in the appropriate section.

### 4.1.1 Preliminaries and Tools

This section covers the tools and algorithms used to create the attack.

**General Image Processing:** General image processing functions are carried out using the Mathworks Matlab 2012a with the image processing toolbox (IPT) 8.0, which together provide a comprehensive set of algorithms and tools for image processing, analysis, visualization, and algorithm development. The toolbox can perform image enhancement, image deblurring, feature detection, noise reduction, image segmentation, and image registration. IPT 8.0 is used in the attack on SQ-PIX for image segmentation and mask generation functions.

**Optical Character Recognition:** OCR plays an important role in gathering data for attacking IRCs. The idea is to use OCR to gather "textual" clues that may be embedded in an image presented for evaluation. While most IRCs keep their keywords in plain text somewhere within the website sourcecode (a poor design choice), clever IRCs turn the text into an image or embed the text into the image itself, thwarting text parsing robots. Some CAPTCHAs can even use a two challenge approach, with one task being a traditional text based CAPTCHA after performing an IRC task. Using OCR as a tool helps to offset this risk to the attacker and provides more information to aid in breaking the CAPTCHA. OCR is utilized in the attacks on ESP-PIX to gather addtional information to increase the probability of a correct challenge response.

**Hierarchical Temporal Memory (HTM) Networks:** HTM networks are a form of neural networks especially suited to image classification. The core component of our attacks on IRCs rely on the use of HTM networks as the primary method of handling image recognition tasks. More specifically, the HTM network gets image input, performs pre-processing on it, and passes the result through multiple levels of processing. Each level builds successively more abstract hierarchical representations, with the highest level representing global image properties and shape. The HTM does this by converting input patterns into sparse distributed representations. Effectively, this means that the image is broken down (the process is termed "sub-sampling" by Numenta)[39] into a fine grid, and each of the grid squares are analyzed separately.

The entire input is reconstructed at the highest layer. An important property of sparse distributed representations is that knowing only a few active bits of a representation is almost as good as knowing all of them. Copies of entire patterns are not stored in the HTM cortical learning algorithm at any time. Learning is based on small subsamples of patterns that, among other things, enable new means of generalization. Sparse distributed representations have many desirable qualities, including robustness to noise, high capacity, and the ability to simultaneously encode multiple meanings. The HTM cortical learning algorithms take advantage of these properties[40]. These representations allow the network to be invariant to small changes in the input and increase the robustness of the system. For categorization tasks, these high-level representations are fed through a supervised classifier at the top of the network. The overall system performs static inference, i.e. there is a single upward pass through the hierarchy. In this network, the first level of coincidences are replaced with Gabor filters of different orientations. At all levels, the coincidence patterns were restricted to have spatial receptive fields smaller than that of the Markov chains.

For our attacks, we build upon this replacement network, utilizing it to process images based on information extracted from the IRC (usually a keyword describing the images). From this output, we can probabilistically perform the task of image identification from the similarity search and image classification done by the HTM network. It is important to note that in the particular implementation used (Numenta Vision ToolKit V1.7.1), there are no feedback connections, temporal inference, or attention mechanisms. All image data larger than 200x200 pixel images is down sampled to this resolution and converted to grayscale images[40]. This downward resolution resampling turns out to be helpful when processing a variety of images from different sources, for the reasons described in detail in the following section.

### 4.1.2 HTM Based Attacks for IRC CAPTCHAs

We now briefly examine each IRC and the challenges they pose to the user. We also describe the procedure for attacking each of IRC based on information that can be gathered from its implementation and design.

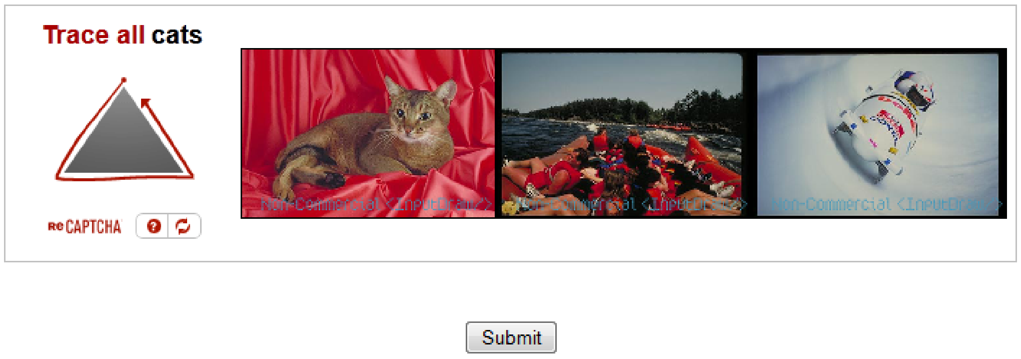


Figure 4.1: Example challenge for SQ-PIX IMAGE:©Carnegie Mellon University

**SQ-PIX** SQ-PIX is an advanced interactive image recognition CAPTCHA that requires the user to trace an outline around the keyword focus in the 3 challenge images. Keyword identification does not need to be handled by OCR, as the word is available in the web source code of the CAPTCHA. Figure 4.1 gives an example of an SQ-PIX challenge. Solving the challenge consists of 3 tasks: keyword identification, image recognition, and tracing the object of interest. Our attack handles each of these tasks step by step, concluding in generating an image mask that provides a defined border around the object of interest. This can then be used to solve the challenge satisfactorily. The attack requires bootstrapping because it must have a minimum of three keywords to build the first HTM networks in order to perform classification. Subsequent HTM networks for different categories can be generated automatically by using the new keyword to gather images from a Bing image search to train and test it. The HTM networks are then used to perform image classification on the three images. Classification is achieved through generation of a probability as to whether the image belongs to *category* $x$ or *not category* $x$, where $x$ is the current keyword and *not* $x$ are all keywords seen excluding $x$ itself. The three images each must be run through all HTM networks to either positively identify them or eliminate them as candidates for selection. The best choice or remaining image after elimination is selected, and passed through an image segmentation algorithm, resulting in a binary image with filled holes that can be used as a mask. The mask is generated using the Sobel edge detection method (both horizontal and vertical detections are performed) to find the border of the object of interest. The outline can then be drawn from the mask border to the CAPTCHA with a custom javascript. If the submitted challenge is successful, the image, mask and the keyword, are saved in a database. Subsequent repeat challenges can be handled via a comparison with the database before this process is initiated again.

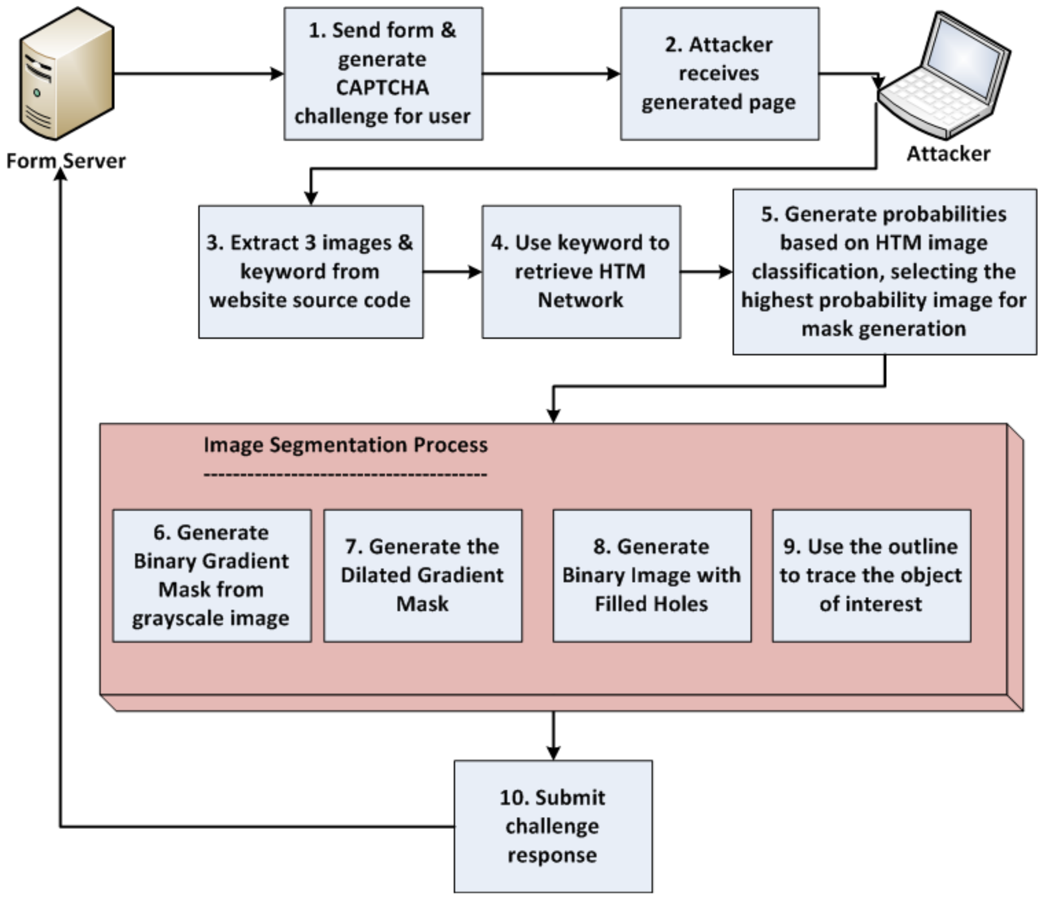

Form Server
1. Send form & generate CAPTCHA challenge for user
2. Attacker receives generated page
Attacker
3. Extract 3 images & keyword from website source code
4. Use keyword to retrieve HTM Network
5. Generate probabilities based on HTM image classification, selecting the highest probability image for mask generation
Image Segmentation Process
6. Generate Binary Gradient Mask from grayscale image
7. Generate the Dilated Gradient Mask
8. Generate Binary Image with Filled Holes
9. Use the outline to trace the object of interest
10. Submit challenge response

Figure 4.2: HTM Based Attack Flow Diagram for SQ-PIX

**The basic SQ-PIX attack is composed of 5 steps:** 1. The attacker extracts the 3 images from the web page source code using the URL's for the images, and the plaintext keyword. 2. The keyword is used to retrieve the (prior built) HTM network used in classification. The networks are generated with images gathered from Bing image search. 3. Use the HTM network to identify the word/image combo via generating probabilities of image likelihood. Take the highest probability image and use that image to start the process for generating the image mask. 4. Use the image segmentation process to "trace" the object of interest. The process allows for either a grayscale image or a binary image. The Matlab script performs the following steps to generate the image mask and the resulting "outline trace" of the object. 4a. Generate the binary gradient mask from the grayscale image. 4b. Generate the dilated gradient mask. 4c. Generate the binary image with filled holes. 4d. Use the binary image with filled holes to draw outline on original image (line in red). 5. The image mask with the outline can then be used to "trace" the object of interest to solve the challenge. If the submitted challenge is successful, the image, mask and keyword are saved in a database. When challenges repeat, time and computational effort can be saved simply by using the matching information in the database. Figure 4.2 provides a flow diagram of this process in action.

**ESP-PIX**

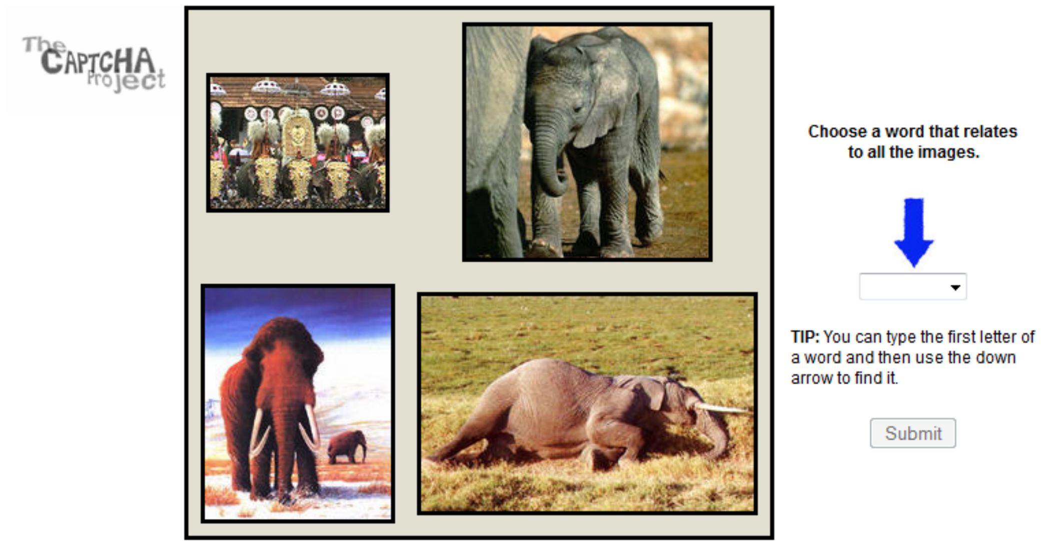

The CAPTCHA Project
Choose a word that relates to all the images.
TIP: You can type the first letter of a word and then use the down arrow to find it.
Submit

Figure 4.3: Example challenge for ESP-PIX IMAGE:©Carnegie Mellon University

In ESP-PIX, the challenge is to select the word from the drop down box list that best describes the four pictures presented in the CAPTCHA frame. This CAPTCHA is unique in that it sometimes uses text to convey the keyword idea, as opposed to an image whose composition and structure correlates strongly with the keyword. This increases the difficulty of a successful attack, because relying solely on HTM networks is not sufficient, since they cannot use text based images. Figure 4.3 gives an examples of ESP-PIX. Attacking this CAPTCHA requires a few tools and some scripting knowledge. The attack relies on a combination of OCR, image recognition algorithms (HTMs), and some heuristics to increase the probability of a successful solution. This combination of tools provides the "widest net" to capture the largest amount of information for determining correct challenge responses. OCR plays a role in deciphering "textual" clues relating to the keyword by scanning the images for text that can be converted to strings. OCR is far from perfect, and frequently produces incomplete strings, or no string at all (cannot convert image text to a string). When this occurs, the attack relies on the best guess from the HTM networks. String manipulation acts as a heuristic, since it is possible to use stemming, Levenshtein distance, and dictionary probability searches to attempt to match any text grabbed by the OCR to a keyword from the list.

Image classification is achieved by using the HTM networks for generation of a probability as to whether the image belongs to *category* $x$ or *not category* $x$, where $x$ is the current keyword starting at the top of the list and *not* $x$ is all keywords from the list excluding $x$ itself. The four images each must be run through all HTM networks to either positively identify them or eliminate them as candidates for selection. Use the probabilities along with the result (if any) from the textual scan and comparison to compute the highest possible probability for the correct response challenge. Submit the response challenge – if accepted, tag the images with the word solution and store all images in the database with the string used to solve the CAPTCHA. Otherwise, discard all information and repeat with a new challenge. Subsequent challenges can simply match the image against the tagged images stored in the database. This increases the speed and accuracy of the attack as time progresses.

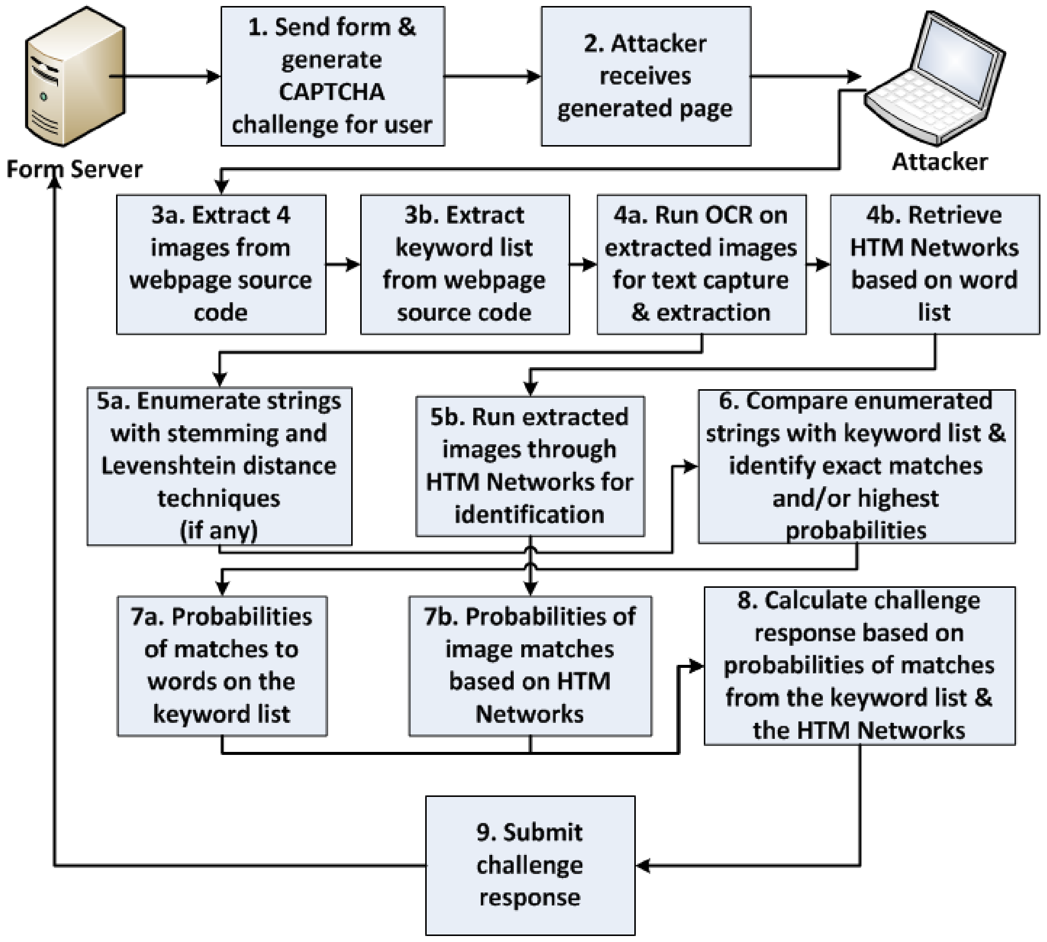
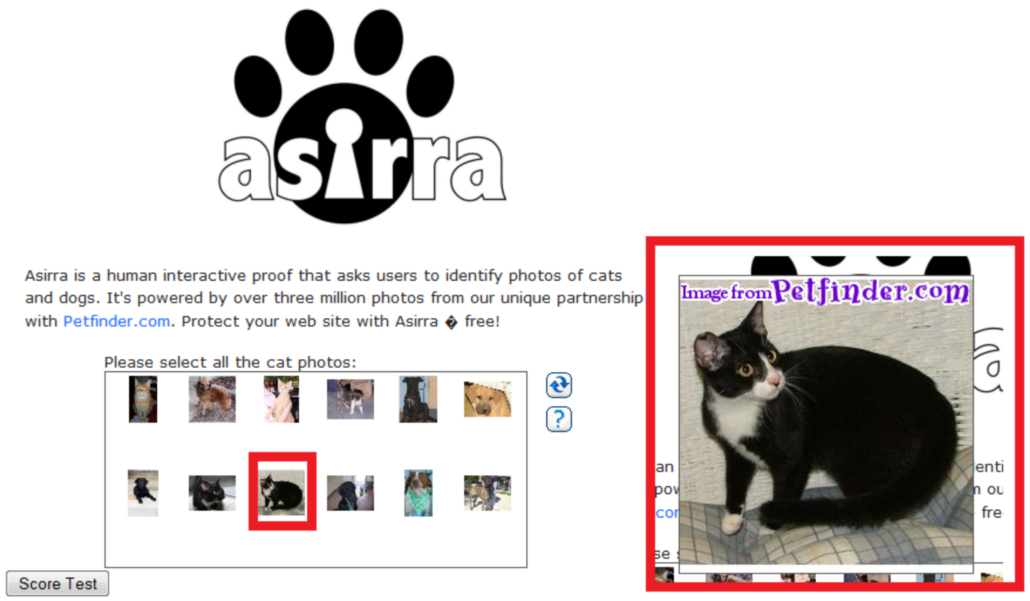

Form Server
1. Send form & generate CAPTCHA challenge for user
2. Attacker receives generated page
Attacker
3a. Extract 4 images from webpage source code
3b. Extract keyword list from webpage source code
4a. Run OCR on extracted images for text capture & extraction
4b. Retrieve HTM Networks based on word list
5a. Enumerate strings with stemming and Levenshtein distance techniques (if any)
5b. Run extracted images through HTM Networks for identification
6. Compare enumerated strings with keyword list & identify exact matches and/or highest probabilities
7a. Probabilities of matches to words on the keyword list
7b. Probabilities of image matches based on HTM Networks
8. Calculate challenge response based on probabilities of matches from the keyword list & the HTM Networks
9. Submit challenge response

Figure 4.4: HTM Based Attack Flow Diagram for ESP-PIX

**The basic ESP-PIX attack is composed of the following 7 steps:** 1. The attacker gathers each of the four images for analysis, along with the list of keywords for the response challenge (These can be extracted from the webpage source code). 2. Generate the HTM networks by using the list of keywords to gather images from Bing image search for training and testing the networks. 3. The four images are each run through an OCR program, to see if any textual data is included in the image that can be extracted for clues to aid in determining the correct challenge response. If no textual clues can be found, step 4 is skipped. 4. Once the textual data is extracted and converted to strings, the list of response challenges is compared to the strings. If there is a match, store the matching word and keep a temporary tag associated with the images. The OCR does not always return perfect results (if any), so edit distance techniques are used to make educated guesses on keywords for the correct challenge response if an exact match is not found. 5. Use the HTM networks previously constructed with images based on the keyword challenge response list to look at the four images. The HTM networks will output the probabilities of matches for each category. 6. Use the probabilities along with the result from the textual scan and comparison to compute the highest possible probability for the correct response challenge. 7. Submit the response challenge - if accepted, tag the images with the word solution and store all images in the database with the string used to solve the CAPTCHA. Otherwise, discard all information and repeat 1-5 with a new challenge. Figure 4.4 provides a flow diagram of this process in action.



Asirra is a human interactive proof that asks users to identify photos of cats and dogs. It's powered by over three million photos from our unique partnership with Petfinder.com. Protect your web site with Asirra � free!

Please select all the cat photos:

Figure 4.5: Example challenge for ASIRRA IMAGE:©Microsoft Research & PetFinder.com)

**ASIRRA** ASIRRA (Animal Species Image Recognition for Restricting Access) is a new HIP whose user challenge revolves around the selection of cats from a set of 12 images composed of cats and dogs drawn from a collection of over 3 million images in the databases at petfinder.com. A correct response to the challenge is to identify all of the cats and submit the answer. Figure 4.5 shows an example challenge for ASIRRA. The reason this CAPTCHA is very strong is that it is quite difficult to tell the difference between cats and dogs, as they visually share many of the same structural traits. In addition, each species and/or breed expresses these traits differently. ASIRRA is unique in that it makes use of an algorithm called PCA (partial credit algorithm) that allows for mistakes as long as the answers being provided for evaluation are *close enough*. Essentially, PCA provides an *intermediate* state, instead of just *correct* or *incorrect*. While the user is solving the CAPTCHA by clicking on the images that are cats, the CAPTCHA is evaluating the responses. From the intermediate state, if the user almost-or completely-solves a subsequent challenge, the user moves to the verified state; otherwise, the user is returned to the unverified state. In ASIRRA, the user moves to the intermediate state if exactly one image (out of 12) is misclassified; from the intermediate state, the user moves to the verified state if zero or one image is misclassified. The ASIRRA attack relies only on HTM networks returning a probability for each picture, to see if the image is a strong candidate for the category “cat” or “dog”. Select the images of cats with the highest probability returned by the HTM network and eliminate dogs by the same process. This method provides the best possible probability for the most accurate guess of which images are cats before the challenge is submitted. Since the number of cats required to be selected varies each time, the fewer the number of cats required to be guessed, the stronger the attack is.

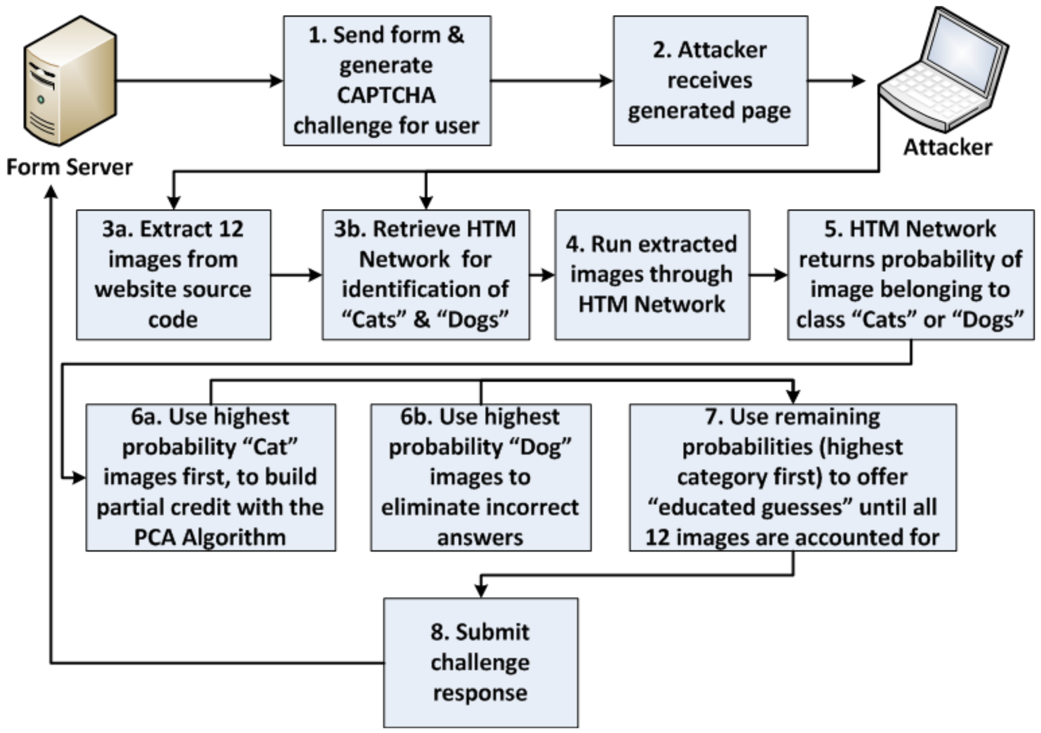

Form Server
1. Send form & generate CAPTCHA challenge for user
2. Attacker receives generated page
Attacker
3a. Extract 12 images from website source code
3b. Retrieve HTM Network for identification of "Cats" & "Dogs"
4. Run extracted images through HTM Network
5. HTM Network returns probability of image belonging to class "Cats" or "Dogs"
6a. Use highest probability "Cat" images first, to build partial credit with the PCA Algorithm
6b. Use highest probability "Dog" images to eliminate incorrect answers
7. Use remaining probabilities (highest category first) to offer "educated guesses" until all 12 images are accounted for
8. Submit challenge response

Figure 4.6: HTM Based Attack Flow Diagram for ASIRRA

**The basic ASIRRA attack is composed of 5 steps:** 1. The attacker extracts the 12 images from the CAPTCHA. 2. Retrieve the HTM network built using the images from the dataset (pick the size that fits your attack best). 3. Run the extracted images through the HTM network for identification. 4. Use the resulting probabilities generated by the HTM network to select the cats with the highest probability (eliminate dogs by the same principle). 5. Use the remaining probabilities to make educated guesses about the remaining images. Figure 4.6 depicts the detailed steps for the attack.

#### 4.1.2.1 Experimental Evaluation

We now discuss the experimental setup and the evaluation results. The experiments are structured to test how well each standalone subsystem of the attacks works at its particular task.

**SQ-PIX and ESP-PIX HTM Generation** The experiments for SQ-PIX and ESP-PIX are structurally similar in that they both use the core methodology of generating HTM networks using search engines to gather images based on keywords provided by the CAPTCHA challenge. There is also some similarity between the keywords in both CAPTCHAs (for example both have cats and dogs as categories). During testing, SQ-PIX revealed 34 different keyword categories while ESP-PIX presents all 72 of its keyword categories at the start.

HTM networks were constructed following the *category x* or *not category x* method for generating probabilities. 50 images were gathered for each *not category x*, giving $33 \times 50 = 1650$ images for SQ-PIX and $71 \times 50 = 3550$ images for ESP-PIX. 25 images in each category are used to train while the other 25 are used to test. This results in 825 images in both training and testing for SQ-PIX and 1775 for ESP-PIX in the *not category x*. The *category x* requires a balanced number of images in comparison, so around 1600 images are needed for SQ-PIX and 3500 for ESP-PIX. This results in 800 images in both training and testing for SQ-PIX and 1750 images for ESP-PIX. Tables 4.1a and 4.1b list the data parameters.

Note that this amount of data is required for testing a single category. The process must then be repeated for every keyword on the list. Since the amount of time required to train and test the HTM networks for all of the categories would be rather large and computationally expensive, we selected *cat* as the *category x* of choice (also since the ASIRRA dataset provided 15,000 images of cats). The remaining image data used in the *not category x* was gathered from a Bing image search using the keyword list. The accuracy of the HTM networks for both CAPTCHAs will be reported along with test cases using the network to identify new cat images, simulating real challenges from each CAPTCHA.

**ASIRRA HTM Generation** The ASIRRA authors provide a large image dataset for public use to crack their CAPTCHA. It consists of a total of 30,000 images in JPEG format with 15,000 images each for cats and dogs respectively. This set is representative of the images from petfinder.com used by the ASIRRA CAPTCHA. However, one caveat is that the dataset does not contain images that would be considered unusable in the CAPTCHA. For example, images that are below a certain resolution, have an aspect ratio that differs too much from 1, or depict animals other than cats or dogs, are all filtered out. Thus, the ASIRRA dataset contains a random, unbiased sample of the images that have passed the acceptance criteria.

The HTM's were created by using the images from the dataset, with a classifier categorizing images as either a cat or a dog. Experiments were set up with 50, 100, 200, 400, 800, 1600, and 12,500 images used for training and testing. The accuracy of the HTM was then recorded, along with the training time taken to generate the network. Table 4.1c gives the data parameters.

The reason for varying the number of images fed into the HTM classifier was to check if increasing the number of images can generate a more accurate representation of the general properties that distinguish a cat from a dog, and vice versa. The reason behind this is because if there is a higher probability of correctly identifying the animal, the probability of beating the challenge posed by the CAPTCHA increases.

Table 4.1: Image data for HTM Network Generation

(a) SQ-PIX (# of Imgs)

| | Category X | Not Category X |
|---|---|---|
| Training | 800 | 825 |
| Testing | 800 | 825 |

(b) ESP-PIX (# of Imgs)

| | Category X | Not Category X |
|---|---|---|
| Training | 1750 | 1775 |
| Testing | 1750 | 1775 |

(c) ASIRRA (# of Imgs)

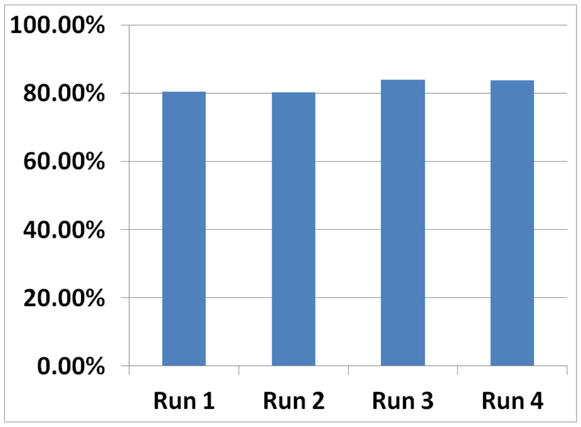
| Group | Category | Training | Testing |
|---|---|---|---|
| 50 Img Network | Cats | 50 | 50 |
| | Dogs | 50 | 50 |
| 100 Img Network | Cats | 100 | 100 |
| | Dogs | 100 | 100 |
| 200 Img Network | Cats | 200 | 200 |
| | Dogs | 200 | 200 |
| 400 Img Network | Cats | 400 | 400 |
| | Dogs | 400 | 400 |
| 800 Img Network | Cats | 800 | 800 |
| | Dogs | 800 | 800 |
| 1600 Img Network | Cats | 1600 | 1600 |
| | Dogs | 1600 | 1600 |
| Final Img Network | Cats | 12500 | 12500 |
| | Dogs | 12500 | 12500 |

#### 4.1.2.2 Experimental Results

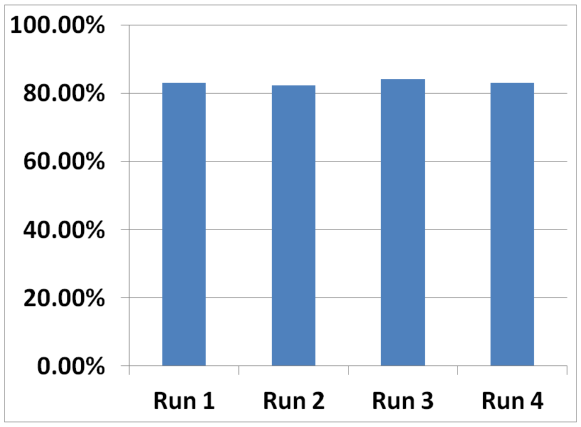
The image datasets discussed before were then used to train and test the HTM networks. In each case, the data was run through 4 test cycles: train & test, which trains the network on the training images, and then checks its accuracy on the test images. This was performed again with the training options turned on, these options include additional training to handle shifts, size changes, mirroring, and small rotations. Finally, two optimization runs were conducted, one with the training options on and one with the training options off. Optimization finds the best set of parameters for the network based on the features found in training images, and then tests the optimized network on the test images for accuracy. Table 4.2 gives the detailed set of system parameters used in each different run.

Table 4.2: HTM Network Generation Parameters

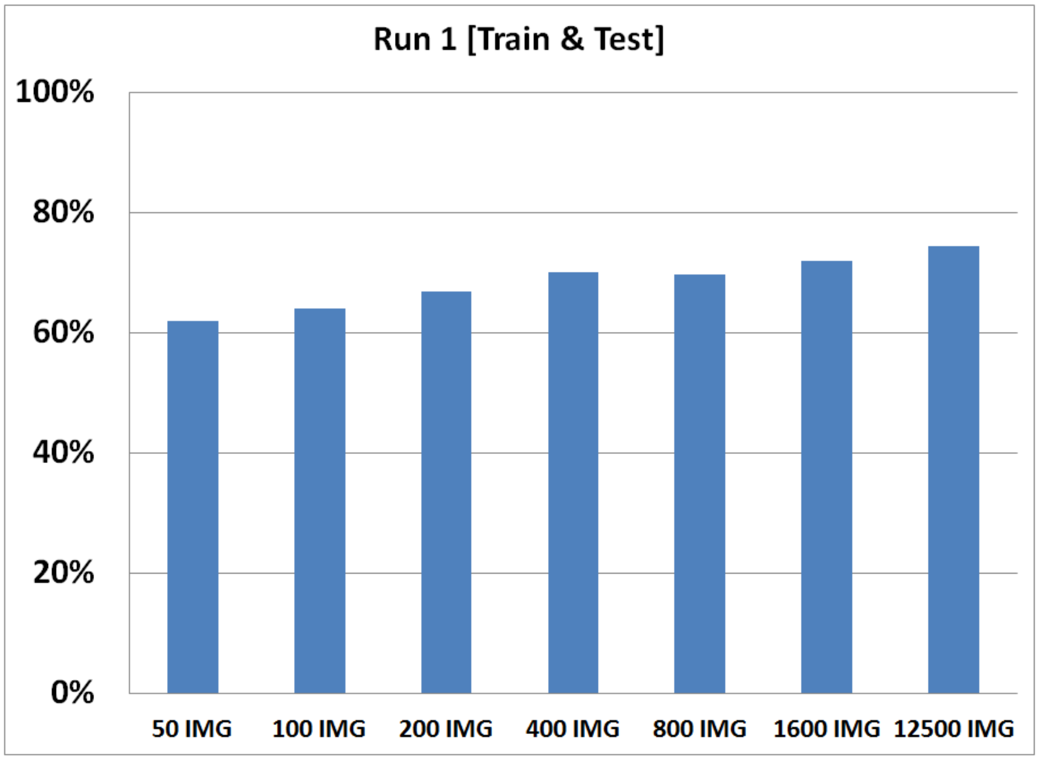
| | Run 1 | Run 2 | Run 3 | Run 4 |
|---|---|---|---|---|
| Action | Train and Test | Train and Test | Optimize | Optimize |
| Shift | n | y | n | y |
| Size Changes | n | y | n | y |
| Mirroring | n | y | n | y |
| Small Rotations | n | y | n | y |



(a) SQ-PIX Accuracy



(b) ESP-PIX Accuracy



(c) ASIRRA Accuracy

Figure 4.7: Experimental Evaluation of HTM Network Attacks on IRCs

Figure 4.7 gives the experimental results. Figure 4.7a shows the accuracy obtained for SQ-PIX. The HTM network has good performance when dealing with distinguishing cats from other images provided by the CAPTCHA. After training for shifts, size changes, mirroring and small rotations, the HTM network achieved 80.3% accuracy. Two additional optimization runs provided a final accuracy of 83.9%. Figure 4.7b shows the accuracy obtained for ESP-PIX. The HTM network has good performance when dealing with distinguishing cats from other images provided by the CAPTCHA, but in this case, having nearly twice as many categories and images as the SQ-PIX HTM network. After training for shifts, size changes, mirroring and small rotations, the HTM network achieved 82.4% accuracy. Two additional optimization runs provided a final accuracy of 83.1%. Figure 4.7c shows the accuracy obtained for ASIRRA for Run 1 (Train & Test). The HTM network has acceptable performance when dealing with distinguishing cats from dogs images provided by the CAPTCHA. The best performance was achieved by the HTM network using 12,500 images, yielding a 74.4% accuracy. However, the 100, 400, and 1600 image networks offer comparable performance at 64%, 70.1%, and 72% respectively, with significantly fewer images. The results for the other runs (2, 3, and 4) were comparable, while taking more time.

Since the HTM provides us with an averaged accuracy for a block of images, some images have a stronger identification probability than others. This means that for a given challenge, images each have a varying degree of probability on being identified as such. The HTM provides a probability for whether the image is a cat or a dog. Our attack revolves around using the cat images with the highest probability of being cats to build a *more correct* answer until the CAPTCHA is solved or we are forced to make an *educated guess* with images that the HTM had a difficult time classifying as cat or a dog. Another benefit is that dogs that can be identified with a high degree of probability can be eliminated from selection, so that the cross section where the HTM cannot tell whether the image is of a cat or a dog (which is where the guess must be made from) is as small as possible.

#### 4.1.2.3 Limitations

While we have obtained results with respect to the category of *cats*, in general, performance may vary as some image categories have a small amount of variation, while others have a wider variety of variation. Thus, the accuracy of detecting the correct category changes with each category and the quality of the images used to train and test the network. The large number of images required in the primary category to maintain balance, (especially in the case of ESP-PIX) can prove to be a challenge to gather, as image search engine results begin to decay rapidly after 1000 images. In this case, it is best to try multiple services and eliminate duplicates to generate a dataset, or to search out labeled datasets that match the category. Cats are on the more difficult end of the spectrum when it comes to detection, thus they made a good choice for selection in the proof of concept HTM networks with regards to network accuracy.

Another point worth noting is that when attempting to use image masks on SQ-PIX, there were several inexplicable failures when tracing the objects of interest. Manual attempts by a real human at tracing produced less than acceptable results, leading to the conclusion that the CAPTCHA has some usability issues. There are also some instances of images misclassified in the CAPTCHA(e.g. a frog in the reptile category) that cause undue failures in the attack. Nevertheless, our results show that the CAPTCHAs are vulnerable to even off the shelf attacks, which can be easily mounted.

### 4.1.3 Web Services Based Attacks for Image CAPTCHAs

This section details the use of various online image processing services that can be used to attack image based CAPTCHAs. Note that these services are constantly evolving and improving, and may have additional capabilities present today that they did not have at the time this research was conducted.

#### 4.1.3.1 Preliminaries & Tools

This section provides some background on traditional image CAPTCHA designs as well as the tools used to create this attack.

**Hard AI Problems for Image CAPTCHAs** Research conducted by Chew and Tygar[13] establish the main characteristics of the challenges presented by image CAPTCHAs that we plan to investigate. While the central challenge to all image recognition CAPTCHAs (IRCs) is a computer vision task (the true hard AI problem), being more nuanced in our descriptions of challenge styles serves to differentiate IRCs from one another. This helps narrow the problem down to manageable areas that are solvable and/or have working CV functions. We primarily focus on three categories for image recognition-based CAPTCHAs, as these are vulnerable to attacks from image based web services. The categories are: 1) Naming images - where the test subject is asked to identify a word associated with a set of images. 2) Distinguishing images - asking the test subject to determine if two subsets of images are associated with the same word or not. 3) Anomaly image - Show 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.

**IRC Exploitable Vulnerabilities & Design Flaws** We would like to emphasize the fact that any and all CAPTCHAs that use images that are indexed by/scraped/acquired from a web image search are potentially vulnerable to our attack. Using an image search to populate a database with images for use in CAPTCHA challenges is a recipe for disaster from a security perspective. It is now trivial to perform a RIS to find a specific image used in CAPTCHA that was generated using the aforementioned method, making solving the challenge much easier than the author intended. Even attempts to distort (crop, color adjust, resize, edited or slightly rotate) the image can be accounted for (to varying degrees of success) by these services. ISS makes finding related images very easy, and if the CAPTCHA presents multiple images from the same word category there exists the possibility that an ISS search will find a number (or potentially all - which can be found by a regular image search once the term is discovered) of the images used to populate a CAPTCHA database. ALA makes it possible to attempt to generate useful information in the form of tags via annotation that can lead to image identification if no other contextual data/metadata can be gathered from RIS or ISS, and offers a last ditch effort to ascertain what is depicted in the image.

One important aspect to the success of these web services is they frequently return filenames within the hyperlink where the image is used on the web. This hyperlink is also accompanied by related text information, often describing the image or a scenario depicted by the image. Since a majority of images are used within the context of the text around them - these two bits of information in particular can give strong clues as to the content of the image. See figure4.10 for an example of this in action. In the case of naming images, it presents the challenge of identifying the correct keyword that describes the series of images. This is best achieved through RIS to attempt to find other places online that use the image in a contextually accurate situation that relates directly to how its being used in the challenge, and ISS to find similarly composed images that fall into a contextual situation that match with the suspected keyword. If no keywords can be found, ALA provides a chance to generate annotations/tags that describe the image and are also potential keywords. When these textual clues are used in conjunction with a related word ontology, either a direct correct answer can be given (direct keyword) or a probabilistic “best guess” can be made. There exist three scenarios for distinguishing images - 1) where the keyword is provided directly,

2) the keyword is provided in a list of keywords, or 3) the keyword is not provided. When these textual clues are used with the keyword ontology, either a direct correct answer can be given (direct keyword) or a probabilistic “best guess” (searching the ontology for related words) can be made. Anomaly images can only be handled by process of elimination - that is - each image needs to be RIS and/or ALA queried and the results evaluated against one another. From this, we can determine the “odd image out” and answer the challenge accordingly. The key idea to remember is that these image based web services make the lookup time for a response very fast. That is, it will generally return a response to a query in well under 30s - a key metric of time as this is considered to be the allowable amount of time for a human to correctly respond to a CAPTCHA. Therefore, any successful bot attack response must be made within this time boundary.

**Related Keyword Ontologies** Related word ontologies provide unique capabilities in that they are designed to find and relate a series of keywords from an image filename or link that could potentially be a correct response to an image challenge. They are designed to use domain knowledge of a subject matter in a meaningful way. For example, if the challenge category word answer is “drinks” and you receive a filename of “350pxtomcollins.jpg” you want to be able to relate that a Tom Collins is a type of drink and thus belongs to the “drinks” category - even though the word “drink” is never explicitly given. The same is true if the link points to a website URL http://www.nationwidebarcrawl.com the association between barcrawls, bars and “drinks” can be discerned by the bot attacker using the ontology (our web of linked data). Utilizing our a priori knowledge of pdictionary, which is a pictoral dictionary of 627 English words designed for illustrations - meaning these words will likely be the keywords used by CAPTCHAs and thus form a good foundation for the core words of the related word ontologies (meaning other words are related to these

627 probable keywords). It is important to note that these ontologies will be organized in a hierarchical structure, as some CAPTCHAs could potentially use words that will be subcategories of a core word. For example, a naming images challenge uses the keyword “horses” - which would be a subcategory of the keyword “animal”. If the images shown have associated URLs such as “thoroughbredhorseranch.com” the ontology needs to be able to know that the word “horse” can be a potential candidate keyword and not choose “animal” as the response. A weighting algorithm designed for word handling can solve this problem. The algorithm computes the number of images used in the CAPTCHA challenge along with the frequency of each word appearing in the return searches from a RIS and an ISS query. So if the keyword “horse” appears in two of the four images’ RIS and ISS results, the probability that “horse” is the keyword and not “animal” is strong. The threshold value for deciding on choosing a core keyword or a subcategory keyword is different for each CAPTCHA, and must be altered accordingly based on a statistical sampling of challenges. In the event that the RIS and ISS turn up no textual clues, the ALA can provide guesses in the form of annotations (tags) as to what the image is depicting. These tags can then be used by the ontology to guess at the context of the image and provide a keyword to solve the challenge. Under these circumstances, a core keyword will always be used.

#### 4.1.3.2 Web Services Attack Methods

This section of the paper serves to provide a generalized framework for which an attack can be created against an image based CAPTCHA using the three aforementioned web services discussed in the preliminaries. Since there exists a wide variety of implementations of image CAPTCHAs, each of which would require its own customized attack parameters and utilities to handle its unique nuances. In this section we will instead focus on the core ideas and tools that enable an attack vector against any implementation that uses indexed images in its generated challenges. The goal is to provide the basis for a viable attack model that can then be modified and/or augmented as needed to defeat the security of an image based CAPTCHA. We leave it up to the attacker to handle the nuances of a particular example.

The basic attack method consists of the following steps: 1) The image based CAPTCHA generates a challenge using indexed images and provides it to the attacker. 2) The attacker receives the challenge and extracts the images from the challenge. 3) The attacker then runs the images through each of the three online tools: RIS (Google& TinEye - each use different indexed databases - so both are required), ISS (Google), and ALA (ALIPR). 4) The search results are recorded from each of the tools - with priority given to RIS results, as this tool attempts to find exact matches and thus the metadata gathered from the results is the most "accurate" of the three. This is then followed by the metadata gathered from the ALA analysis of the image and finally the ISS last. 5) The metadata from the RIS and the ALA can be analyzed for any potential keyword matches, as this signifies a strong probability of the what is depicted in the image. 6) From this metadata, a probabilistic guess can be made for the response to the challenge. This requires the use of the custom generated ontologies that aid in handling keyword relationships for more accurate guesses. If the guess is correct, save the images and the keyword(s) in a database for reuse in the future.

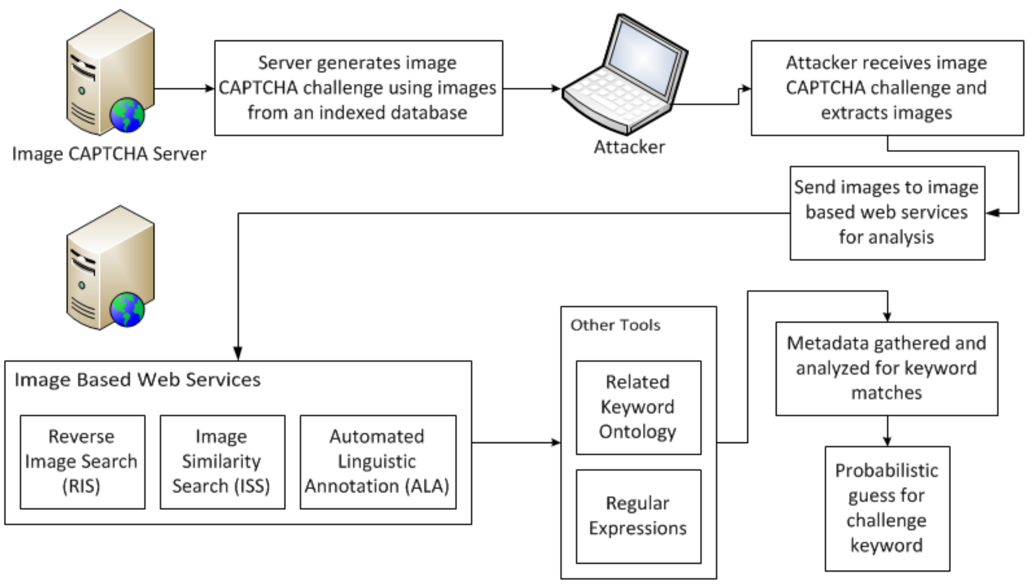
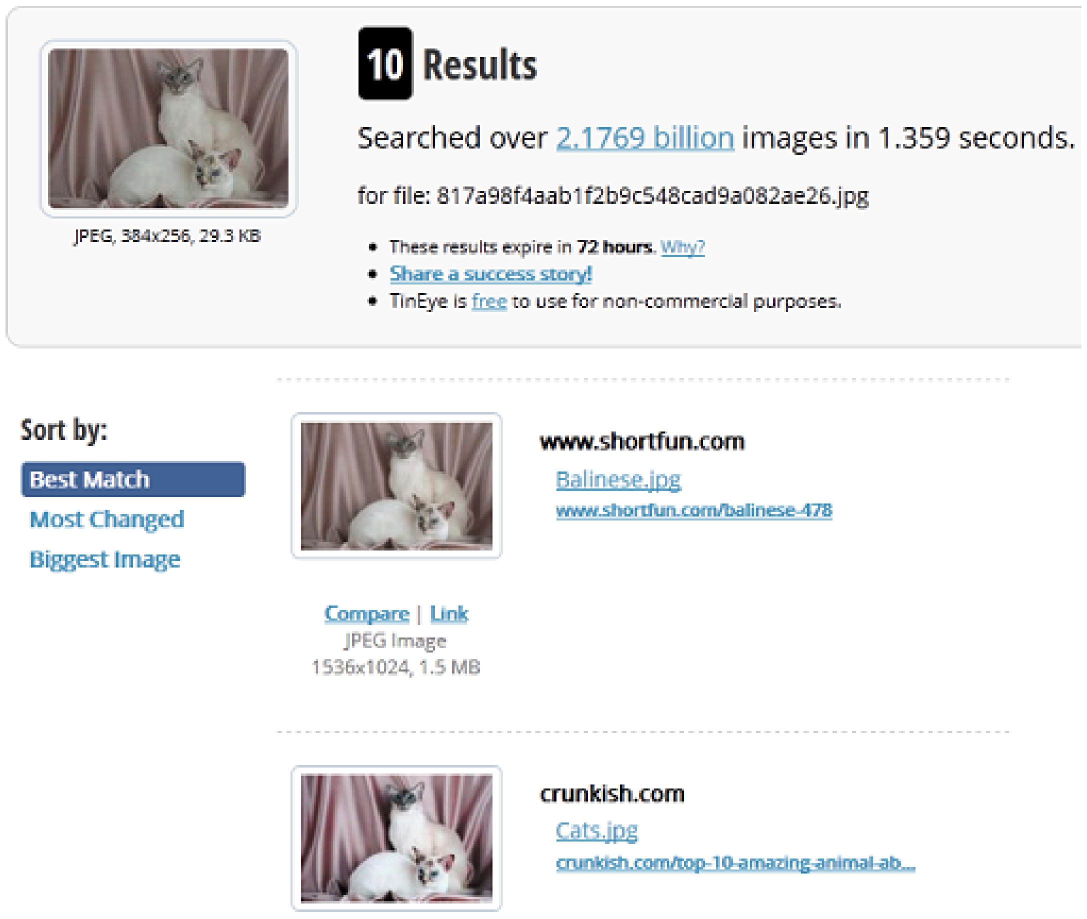


Image CAPTCHA Server
Server generates image CAPTCHA challenge using images from an indexed database
Attacker
Attacker receives image CAPTCHA challenge and extracts images
Send images to image based web services for analysis
Image Based Web Services
Reverse Image Search (RIS)
Image Similarity Search (ISS)
Automated Linguistic Annotation (ALA)
Other Tools
Related Keyword Ontology
Regular Expressions
Metadata gathered and analyzed for keyword matches
Probabilistic guess for challenge keyword

Figure 4.8: General Attack Method for Image Based Web Services

To demonstrate the effectiveness of the online image based tools, a sampling of images from popular image based CAPTCHAs were taken to see what types of metadata could be gathered in order to get around the requirements of performing an image classification task. The only assumption the attacker must make is that the CAPTCHA is using images that are indexed online. If the metadata gathered from the image search is descriptive enough, this information can be used to “solve” the classification task.

### Reverse Image Search



10 Results

Searched over 2.1769 billion images in 1.359 seconds.

for file: 817a98f4aab1f2b9c548cad9a082ae26.jpg

JPEG, 384x256, 29.3 KB

- These results expire in **72 hours**. Why?
- Share a success story!
- TinEye is free to use for non-commercial purposes.

Sort by:

- Best Match
- Most Changed
- Biggest Image

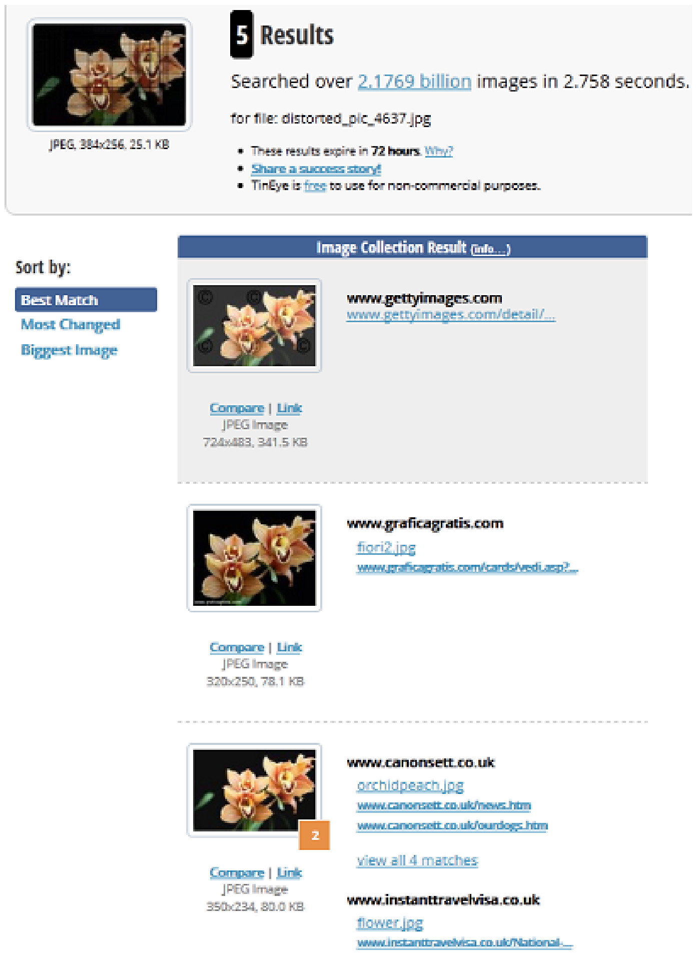
**www.shortfun.com**
Balinese.jpg
www.shortfun.com/balinese-478

Compare | Link
JPEG Image
1536x1024, 1.5 MB

**crunkish.com**
Cats.jpg
crunkish.com/top-10-amazing-animal-ab...

Figure 4.9: RIS results w/metadata for an extracted image IMAGE: ©TinEye - Idee Inc.

Figure 4.9 shows an example result from a Reverse Image Search (TinEye) done on an image of “cats” provided by an SQ-PIX challenge. The power of the “exact” match is that RIS tells you where else on the web the image is being used. It also provides the metadata that accompanies the use of that image (filename, URL, image properties etc.). Using the “best match” functionality, the second hit in a list of exact matches has a descriptive filename, “cats.jpg” - a match to the challenge keyword of “cats”. Note that the file submitted by the attacker has a long and obfuscated filename (a straight extraction from the challenge HTML) and is a different size than the resulting matches. The first hit is actually too accurate - the proper name for the breed of cat depicted in the image - but this requires a related word ontology to be able to discern that "Balinese" is indeed a breed of cat. From our experimentation, SQ-PIX is believed to have approximately 34 different keyword categories from which challenges are generated, although we have no way of knowing the exact number of keywords - this figure was derived from a statistical sampling. The accompanying word ontology can then be generated from this list of keywords with some effort on behalf of the attacker.



JPEG, 384x256, 25.1 KB

5 Results

Searched over 2.1769 billion images in 2.758 seconds.

for file: distorted_pic_4637.jpg

- These results expire in **72 hours**. Why?
- Share a success story!
- TinEye is free to use for non-commercial purposes.

Sort by:
Best Match
Most Changed
Biggest Image

Image Collection Result (info...)

**www.gettyimages.com**
www.gettyimages.com/detail/...

Compare | Link
JPEG Image
724x483, 341.5 KB

**www.graficagratis.com**
fiori2.jpg
www.graficagratis.com/cards/vedi.asp?...

Compare | Link
JPEG Image
320x250, 78.1 KB

**www.canonsett.co.uk**
orchidpeach.jpg
www.canonsett.co.uk/news.htm
www.canonsett.co.uk/ourdogs.htm

view all 4 matches

2

Compare | Link
JPEG Image
350x234, 80.0 KB

**www.instanttravelvisa.co.uk**
flower.jpg
www.instanttravelvisa.co.uk/National-...

Figure 4.10: RIS results w/metadata for obscured image IMAGE: ©TinEye - Idee Inc.

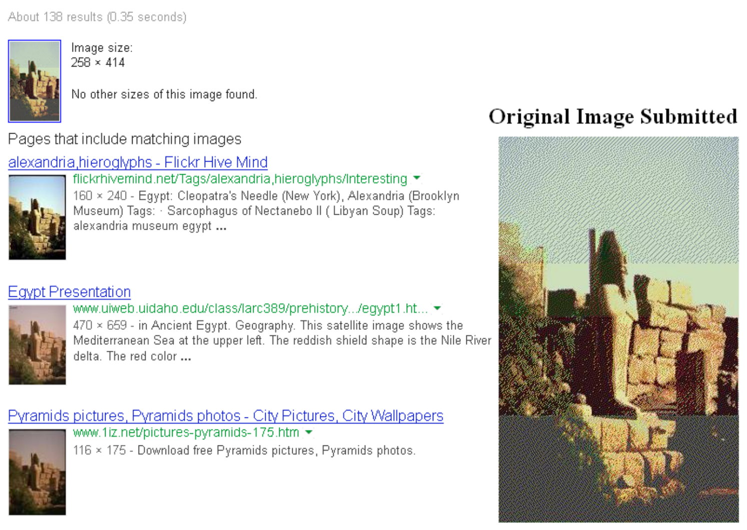

About 138 results (0.35 seconds)

Image size:
258 × 414

No other sizes of this image found.

Pages that include matching images

alexandria,hieroglyphs - Flickr Hive Mind
flickrhivemind.net/Tags/alexandria,hieroglyphs/Interesting
160 × 240 - Egypt: Cleopatra's Needle (New York), Alexandria (Brooklyn Museum) Tags: · Sarcophagus of Nectanebo II ( Libyan Soup) Tags: alexandria museum egypt ...

Egypt Presentation
www.uiweb.uidaho.edu/class/larc389/prehistory.../egypt1.ht...
470 × 659 - in Ancient Egypt. Geography. This satellite image shows the Mediterranean Sea at the upper left. The reddish shield shape is the Nile River delta. The red color ...

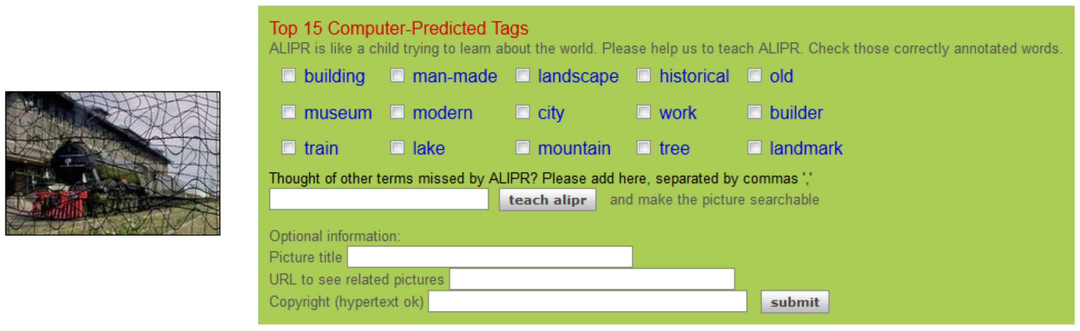
Pyramids pictures, Pyramids photos - City Pictures, City Wallpapers
www.1iz.net/pictures-pyramids-175.htm
116 × 175 - Download free Pyramids pictures, Pyramids photos.

Figure 4.11: RIS Results w/ obfuscated image - IMAGINATION Stage 1 IMAGE: ©Google Image Search

There have been attempts by some CAPTCHA developers to obstruct and obscure their images with noise to foil attackers that attempt to use image classification and object recognition tools to discern what is depicted in the image. Unfortunately for the developers, Tineye is robust against alterations to images, and can find exact matches of images that have been obstructed, obscured, or altered in a meaningful way. In Figure 4.10 the image used in this challenge was generated by the IMAGINATION CAPTCHA[18] - which is a two stage CAPTCHA. IMAGINATION uses obfuscation of its images as an added layer of security from automated image recognition/object recognition attacks. The image in the example comes from the second stage, which is a image classification/object recognition challenge - asking the user to match the image with the keyword from a list. The first stage has a compelling attack developed by Zhu et. al.[78], but none have been provided for the second stage except this one. For an example of the second stage IMAGINATION challenge, see figure 4.13 in the following section.

It is clear from these experiments that Reverse Image Search is a powerful tool that can be used by an attacker to gather important information relating to image depictions used in CAPTCHA challenges. Based on these examples, we recommend that designers take note of the possibility of this attack vector and take steps to protect their image based CAPTCHAs from this tool in particular.

**Automated Linguistic Annotation** Automated Linguistic Annotation is a useful tool when the metadata gathered by RIS and ISS does not provide any easily distinguishable clues as to the content contained in the challenge images. Since it uses advanced algorithms to discern what is depicted in the image, an attacker uses it as a last resort to attempt to generate some metadata that may be relevant and lead to the ability to generate a guess against a list of keywords, or just provide a slightly more accurate random guess. Since we are using ALIPR as the tool for ALA, it generates 15 tags based on an analysis of the image provided to it. These tags can then be saved and compared against a keyword list from the CAPTCHA. Figure 4.13 is an example of automated linguistic annotation. One additional benefit is that images submitted to ALIPR are saved to its index, so that over time it will have more images with correct tags and more accurate tags provided by humans - allowing for an increase in accuracy when used by an automated service like this attack bot.



Top 15 Computer-Predicted Tags

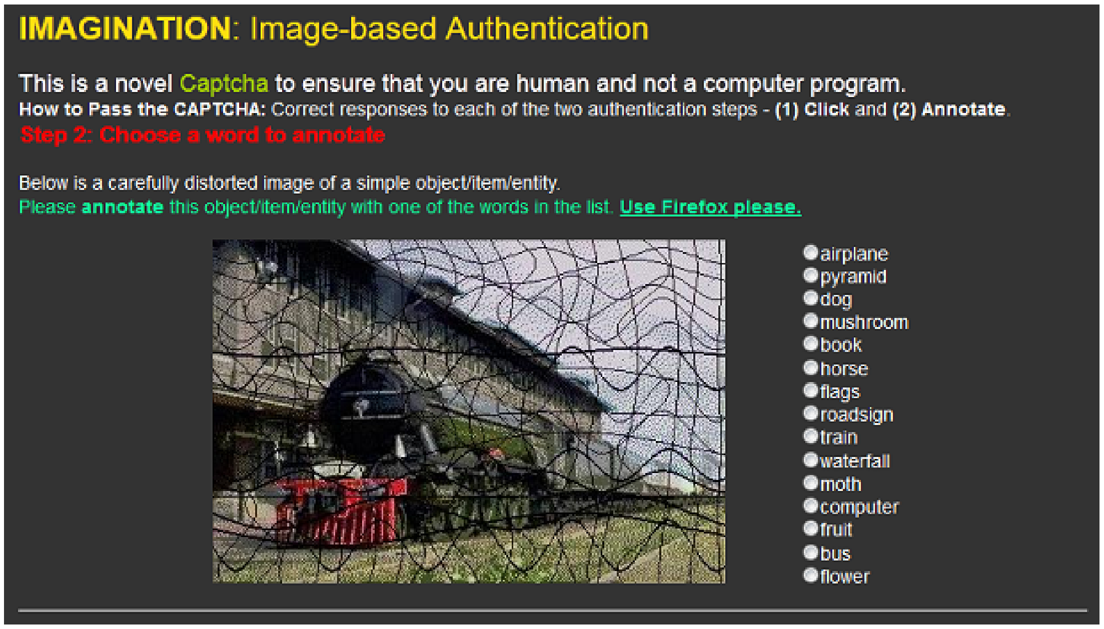
ALIPR is like a child trying to learn about the world. Please help us to teach ALIPR. Check those correctly annotated words.

| | | | | |
|---|---|---|---|---|
| ☐ building | ☐ man-made | ☐ landscape | ☐ historical | ☐ old |
| ☐ museum | ☐ modern | ☐ city | ☐ work | ☐ builder |
| ☐ train | ☐ lake | ☐ mountain | ☐ tree | ☐ landmark |

Thought of other terms missed by ALIPR? Please add here, separated by commas ','

Figure 4.12: ALA results generate 15 tags - exact match “train” IMAGE: ©ALIPR - Penn State University

It is also worth noting that ALIPR is somewhat robust against noise interfering in the tag generation process. The image from the example was submitted for evaluation and it was still able to generate tags for many of the key features contained in the image, including the keyword, “train”.



**IMAGINATION**: Image-based Authentication

This is a novel Captcha to ensure that you are human and not a computer program.
**How to Pass the CAPTCHA:** Correct responses to each of the two authentication steps - **(1) Click** and **(2) Annotate**.
Step 2: Choose a word to annotate

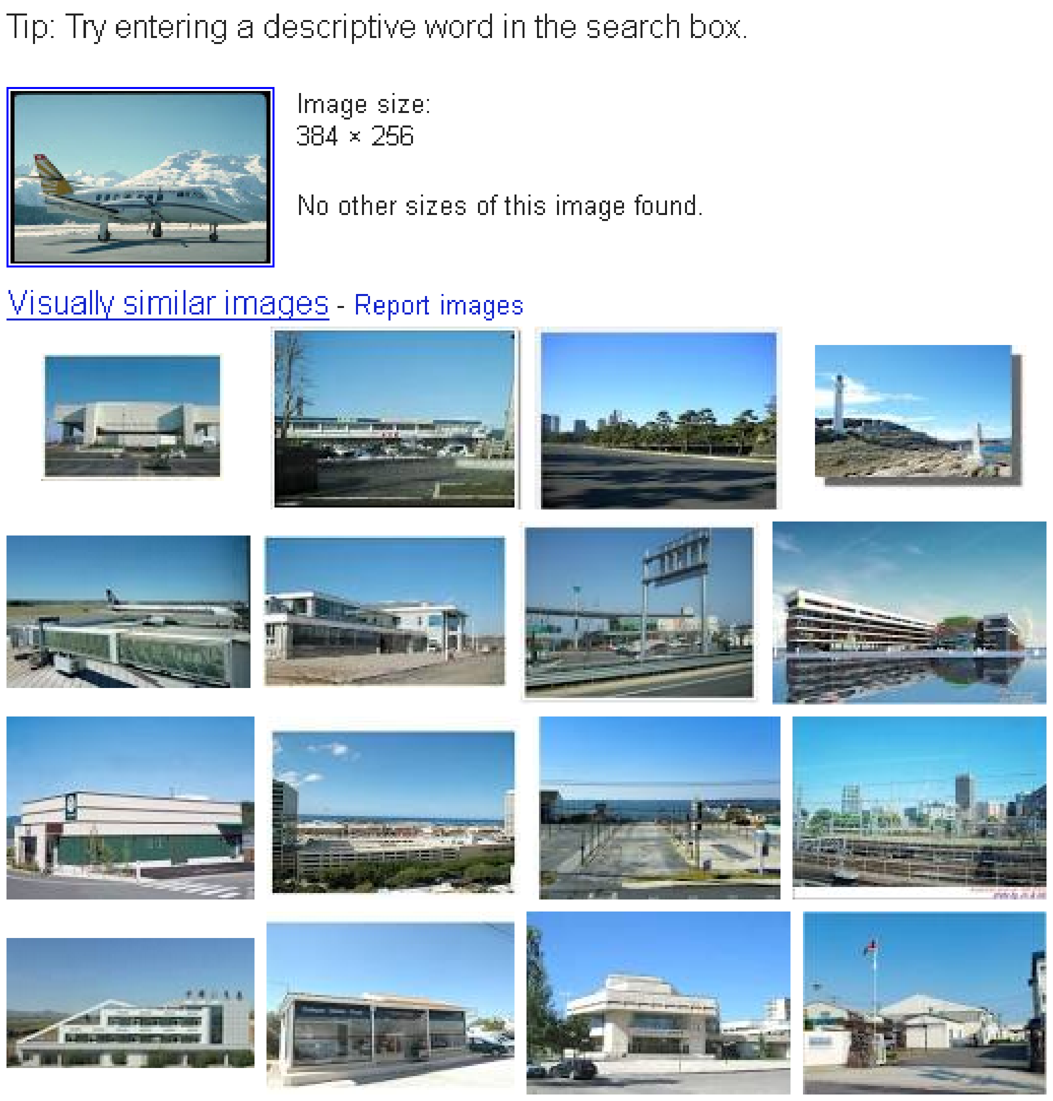
Below is a carefully distorted image of a simple object/item/entity.
Please **annotate** this object/item/entity with one of the words in the list. **Use Firefox please.**

- airplane
- pyramid
- dog
- mushroom
- book
- horse
- flags
- roadsign
- train
- waterfall
- moth
- computer
- fruit
- bus
- flower

Figure 4.13: Example second stage challenge for IMAGINATION IMAGE: ©Penn State University

While this seems like a very powerful approach to solving the problem - indeed from a surface analysis it appears to be a better solution than RIS, it is not without its shortcomings. ALA is not very accurate - most of the tags generated are too broad to be of use. Li and Wang[47], the developers of ALIPR, state that the goal of their automatic annotator is to be able to provide 98% of images with at least one correct annotation out of the top 15 tags generated. The highest ranked annotation word for each image is accurate with a rate above 51% (in their experiments with image data from FLICKR). Knowing this fact, ALA is best used along with RIS for comparison of metadata and as a generator of metadata when RIS or ISS fails - never by itself exclusively. However, this research is very interesting and it is easy to imagine that it will improve in speed and accuracy with time, which should give pause to CAPTCHA designers who wish to use image recognition/object recognition as a challenge.

**Image Similarity Search** Google currently provides the capabilities to upload images to its search engine and returns images that are similar in color and composition to the uploaded image. This is useful for attempting to discern information that can be derived based on some conjectures and statistical sampling. For example, if the image has a unique, specific structure to it (e.g. a car, a bus etc.) it is likely that similar images featuring these same properties will show up - this is the composition component of ISS. The color component plays different role, that of discerning broad generalities about the image - e.g. if a certain percentage of the image contains large amounts of certain shades of green or blue pixels - there is a probability it could be sky, water, grass etc. or if an image contains skin tone shades within a specified range for people. In addition to this - the metadata for the top results can be gathered and analyzed against information from the CAPTCHA. In figure 4.14 in the top 16 results - there is one image featuring an airplane. Depending on how much information the CAPTCHA provides, a high probability guess can be wagered if used in conjunction with the related keyword ontology.



Tip: Try entering a descriptive word in the search box.

Image size:
384 × 256

No other sizes of this image found.

Figure 4.14: Google ISS Results - Keyword Airplane IMAGE: ©Google Image Search

Image similarity search also works to a degree with obscured images. However, since the similarity algorithm looks at the composition of the image, the distortions and obfuscations play a role in influencing what images it returns as results. In figure 4.15 the strong features of the grid have influenced the results - however another bus image is still seen in the top 16 results.

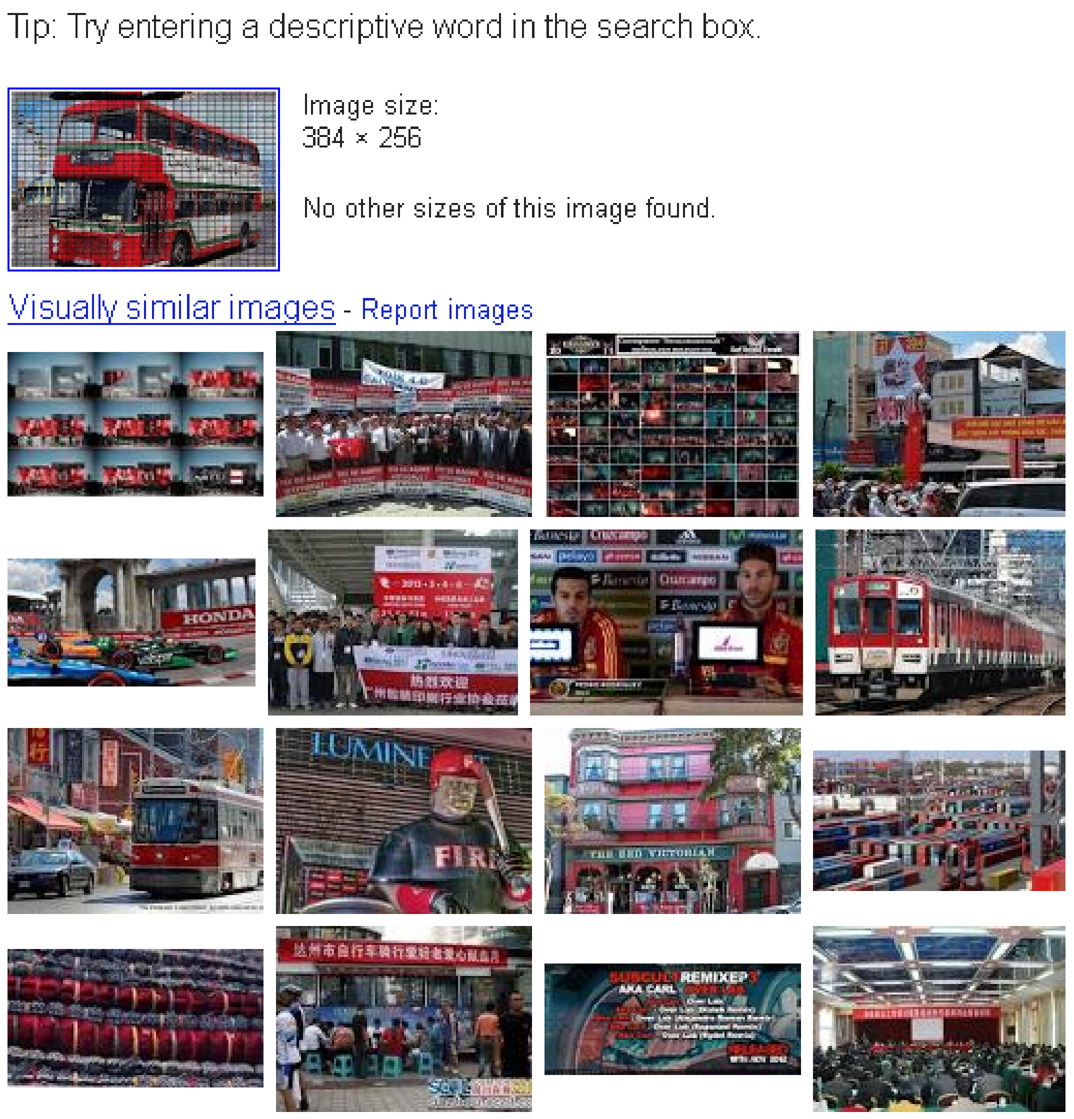

Tip: Try entering a descriptive word in the search box.

Image size:
384 × 256

No other sizes of this image found.

Figure 4.15: ISS Results w/ obfuscated image IMAGE: ©Google Image Search

### 4.1.3.3 Experimental Results & Analysis

In this section we present the results of a few brief experiments to demonstrate the capabilities of the image based web services in action on real world image CAPTCHAs. However, we proceed with a few caveats. Unfortunately, many of the CAPTCHAs (ESP-PIX, SQ-PIX, IMAGINATION) and some of the tools (ALA) are dead - as their web sites time out, have broken image links or are in some other way unusable before we could finish authoring this paper. In addition to this, most of the CAPTCHAs described in the literature have never been deployed publically or are unavailable for evaluation. If the authors of the CAPTCHA explicitly mention their method to populate their image database, it is probably safe to assume that similar results to the ones we do provide would be achievable by a clever attacker. The greatest loss is that of the use of ALIPR - we have no other online ALA tool and therefore cannot report any meaningful results to its use beyond the small number of test cases we performed for the methodology examples. Fortunately, we did manage to save 82 unique images from SQ-PIX challenges, 11 distorted first stage images and 13 distorted second stage images from IMAGINATION while they were still online. We also include a control group of images that are not indexed / available on the web. The control group images are immune to RIS and can only be attacked by ISS and ALA. We would like to make note that the composition of the image directly influences the results of the ISS e.g. if you take a picture of a cat that is focused primarily on the cat, ISS results will have pictures of cats in the top 16 results. However, if you compose the shot so that the cat is part of a larger scene, the results will vary greatly and most likely will not return an image of a cat in the results (cat is small percentage of overall composition of the image).

Our two selected CAPTCHAs in particular serve the evaluation purposes sufficiently. SQ-PIX images serve as the clear/free of distortions challenge, and IMAGINATION images serve as the distorted/altered challenge. While we do offer results, they are to be taken with a grain of salt. The sample size we used is limited and there exists the possibility that the challenges we received were “easy” in comparison to what would have been generated in a larger sample. Nevertheless, we believe the method is strong and our results validate the argument that any and all CAPTCHAs that use images that are indexed by/scraped/acquired from a web image search are potentially vulnerable to attacks from image based web services.

| CAPTCHA Type | RIS(Google) | RIS (TinEye) | ISS (Google) | Total |
|---|---|---|---|---|
| SQ-PIX | 90% | 60% | 30% | 90% |
| IMAGINATION Stage 1 | 25% | 25% | 12% | 37.5% |
| IMAGINATION Stage 2 | 30% | 30% | 12% | 38% |
| Control Images(Not Online) | 0% | 0% | 20% | 20% |

Table 4.3: Image Based Web Services - Accuracy Results

#### 4.1.3.4 RIS Results

Using our limited data, these were the results generated. Your miliage may vary. We encourage attackers and design authors alike to test out the tools in the manner we suggested in the methodology.

**IMAGINATION** Out of 11 images for RIS on stage 1 - with each image consisting of 8 smaller images that must be tested individually after segmenting them (88 imgs total), on average, tineye found 2/8 and google found 2/8 (frequently they would share the same hit or all hits - each hit can only be counted once), thus the average combined total for RIS is 25%. Out of 13 images total for RIS on stage 2, tineye found 4/13(cat, tiger, flower, peppers) and google found 4/13(cat, tiger, flower, train) thus the combined total, counting identical matches once (cat, tiger, flower, peppers, train) for RIS on stage 2 would be 5/13=38.4%.

**SQ-PIX** Out of 82 unique images - google was able to find 74/82=90% of the images we sampled. Tineye was able to find 49/82=60% of the images we sampled. In our test, every miss was a miss by both tineye and google but every match for tineye was also a match for google. Thus we end up with an average accuracy of 90% for the SQ-PIX CAPTCHA.

**ISS Results** The ISS results are based on whether or not a meaningful result (i.e. a similar image that was accurate - e.g. a cat picture provides more cat pictures and not landscapes) could be given from the top 16 hits returned. In the case of

SQ-PIX, about 24/82=30% images had meaningful results in the top 16 hits. For IMAGINATION, stage 1 had a 1/8=12% and stage 2 had a 1/8=12% meaningful result. ISS was more useful on stage 2 than on stage 1, as stage 1 needs an exact match if there is any hope of finding the geometric center of the image - in this case a similar image is useless (the challenge solution to stage 1). One shortcoming was that almost every meaningful result from ISS was with an image that also had a match from RIS for both SQ-PIX and IMAGINATION - thus the ISS is not as useful as it seems on the surface.

#### 4.1.3.5 Limitations & Effective Defensive Measures

There are a number of limitations in using these tools to mount an effective attack. However, some of these shortcomings affect both attackers and the designers of CAPTCHAs. The following list contains the major points for consideration:

- Non-Indexed images - RIS cannot provide any help if the images in question are not part of its index. Relying on ISS and ALA is not as strong as having identical matches and their related information to work with, and as such these images provide strong security against these attack vectors.

- Service Throttling - Free image based web services have a finite amount of times a user is able to submit queries without paying a fee for premium service or having their IP banned or throttled for a period of time. This is intentional in design to discourage abuse of the systems by bots, ironically enough.

- Composite images w/ distortions - composite images provide a challenge for RIS as they do not have an image fingerprint to find. While they are composed of other images that may be indexed, finding the original images can prove to be a difficult - if not impossible - challenge. When distortions and obfuscations are added this makes it even more difficult. As such, these images provide strong security against RIS and ISS. However, if the image is too ”unnatural” in its composition in that the images added to the base image stand out easily, they can be identified by ALA or segmented out the image and analyzed individually. In this case, they provide weak security in the image recognition context.

# Chapter 5

# Defense of CAPTCHAs

This section covers the defensive methods that we have created to make attacks on image based CAPTCHAs more difficult.

## 5.1 Enhancing Security of Image CAPTCHAs through Noise Addition

This section details the work conducted on using noise to prevent images from being recognized by RIS and CV attack algorithms for the express purpose of being used in an image CAPTCHA.

### 5.1.1 Defense Strategies for IRCs

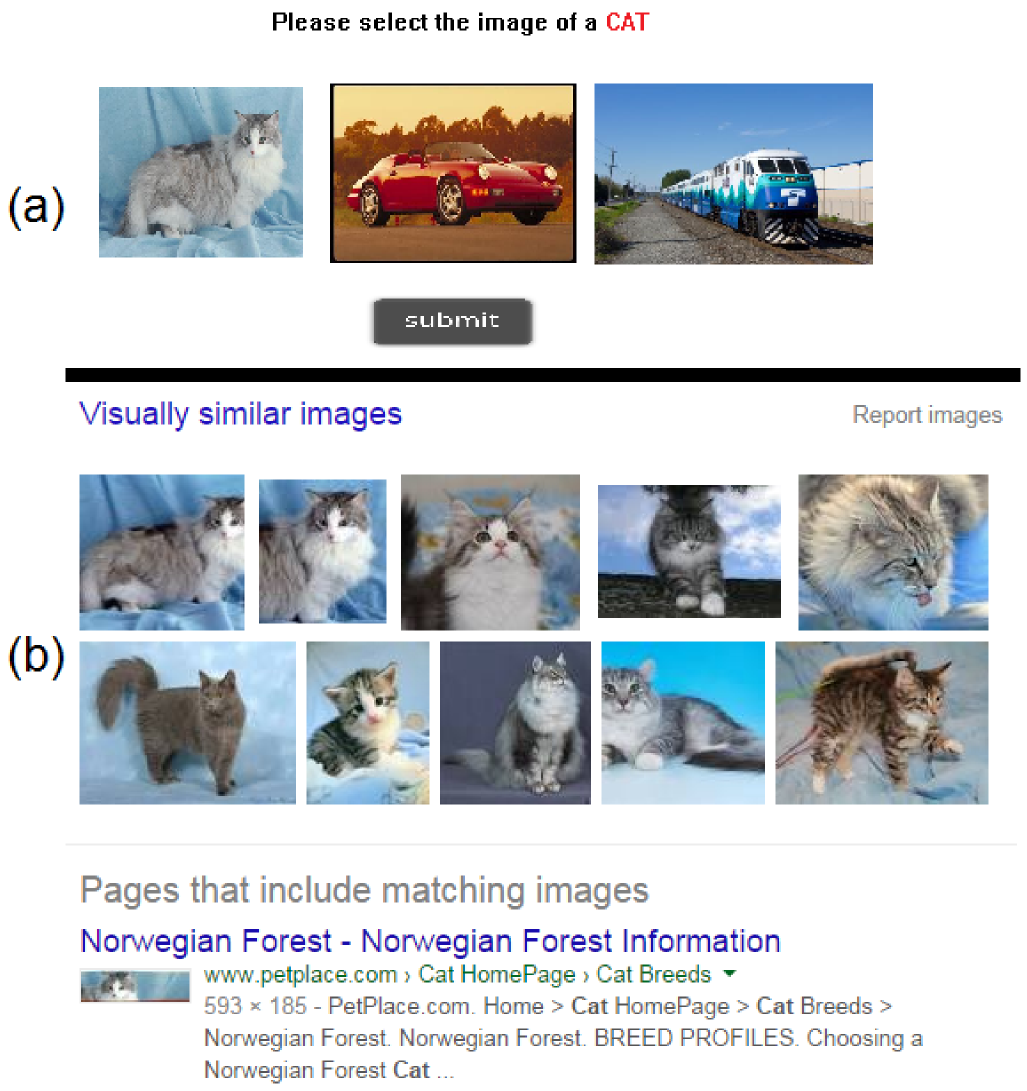
The goal of our work is to address several of the security shortcomings of image CAPTCHAs, and to solve them with a generic approach of adding noise to the image. We now discuss two particular types of attack – Reverse Image Search (RIS) engine attacks and Computer Vision (CV) attacks. These are particularly strong against image CAPTCHAs. Note that although we discussed an RIS attack in the previous section, by the time this research was performed (Mid-Late 2014), significant upgrades to RIS tools had been made by their companies respectively. For example, Google image search had started to provide a "guess" as to what the image was depicting in the form of a text string, and Tineye had significantly increased the size of its search image database (several billion more images). At the time this research was performed, the Google keyword guess was not very accurate, however it is constantly improving over time and presents a legitimate and accurate threat as of today (Mid 2016). We also discuss the general defense strategy of adding noise to the image to make it more robust to these types attacks. As with all CAPTCHAs, we are again faced with the challenge of balancing security with usability. Since we are utilizing a noise addition method, the image cannot be altered to the degree that a human observer loses the ability to recognize the content of the image (rendering it useless for our purposes).

**Stopping Image Search Attacks** First, our noise addition algorithms must stop reverse image search engines from finding image matches indexed online (Google image search[1] and Tineye [2]). This is an important security enhancement as image CAPTCHAs traditionally have problems in defending against database attacks and tag matching attacks, which can be viewed as a scalability issue (too few unique images). The following scenario is an example of an RIS attack in action: Imagine an image based CAPTCHA challenge asking the user to identify which image out of a set of images depicts a cat as shown in Figure 5.1. The attacker then: 1) Makes a copy of the images from the CAPTCHA, 2) Runs them through an RIS engine to find exact matches, 3) Scrapes and stores the metadata from the RIS engine, 4) Uses Regular Expressions to match the keyword "cat" to the search that locates a copy of the image used somewhere else online with the filename "cat.jpg", which happened to be found on a website with the URL that contains the word "cat" e.g.

[1]https://www.google.com/img
[2]https://www.tineye.com/

http://www.coolcatpics.com.

At this point, the attacker can probabilistically determine which image is most likely the cat image (or eliminate the other image choices through the same process). The bad news for those attempting to develop security measures against RIS engine attacks is that the engines themselves are proprietary (trade secret) and closed source, forcing the CAPTCHA security developer to devise a set of experiments that attempt to probe a “black box” to learn its behavior. The good news is that the RIS engines are available for use by the public with reasonable limits established (50 test images per day, up to 150 per week), and a security expert with access to or knowledge of “image fingerprinting” and image processing literature can use this body of knowledge to provide clues for educated guesses as to the methods that RIS engines are utilizing to identify matches. The noise generation method we propose works on the premise of introducing an amount of noise such that the image used for a CAPTCHA challenge has been altered enough from the original that the various “image fingerprinting” metrics used to determine matches have been “tricked” - that is they no longer see the image as a match as its information diverges from the original image beyond their threshold/similarity metric. Technically speaking, the image returned by the method is a different image, as the noise changes the values of the pixels in the image. A distance metric (change from original) is useful to model the noise alterations from original image to new image. However, the new image (post-noise) is still functionally depicting the same content as the original, albeit in a degraded fashion. Stopping RIS engines from finding matches means indexed images can be used as CAPTCHA challenges again, increasing the sample space of potential usable images significantly.


Please select the image of a CAT
(a)
submit
Visually similar images
Report images
(b)
Pages that include matching images

Norwegian Forest - Norwegian Forest Information
www.petplace.com › Cat HomePage › Cat Breeds
593 × 185 - PetPlace.com. Home > **Cat** HomePage > **Cat** Breeds > Norwegian Forest. Norwegian Forest. BREED PROFILES. Choosing a Norwegian Forest **Cat** ...

Figure 5.1: Reverse image search attack with metadata. (a) depicts the CAPTCHA images without noise, (b) depicts results of a Google image search IMAGE: ©Google Image Search

**Stopping Computer Vision Attacks** Second, the noise algorithms must be able to alter the image enough to hinder or stop altogether, general image/object recognition algorithms that would attempt to solve image recognition challenges.

One popular CV algorithm is SIFT [51], which stands for Scale-Invariant Feature Transform. While it has previously been used in many applications, we are interested only in its ability to perform object recognition tasks. ASIFT [59], which stands for affine-SIFT, is an improvement over SIFT. It considers the lattitude and longitude angles that are ignored by SIFT and then combines that information with SIFT to provide a more complete analysis than SIFT alone. As such, it significantly outperforms SIFT and is more of a challenge to defeat. By adding noise to the image, it should throw off the keypoints calculations so that when it compares two images, the noised image does not have the same keypoints and it fails to return a match. Note that the web application uses grayscales and resizes the images before the CV algorithm is run.

Another important point to consider is that we used an online service to perform the SIFT and ASIFT analysis [59]. The above computation could be completed in approximately 7 seconds through a web form. As more of these services move online, an attacker no longer needs to run local image matching or CV tools, and can script a live attack that pipes the CAPTCHA challenge through the appropriate tools to generate and even submit a correct response. To see the breadth and depth of online CV tools available to attackers, Carnegie Mellon University's Calibrated Image Lab[45] hosts a web portal with links to various online CV tools. While many of the links are broken/dead, the fact that they exist means that a CAPTCHA designer must take into account the potential risk of an attacker reading the literature and re-implementing the method.

For example, in our aforementioned cat image scenario, imagine in this case the attacker decides to use image/object recognition with a CV toolkit. The attacker has

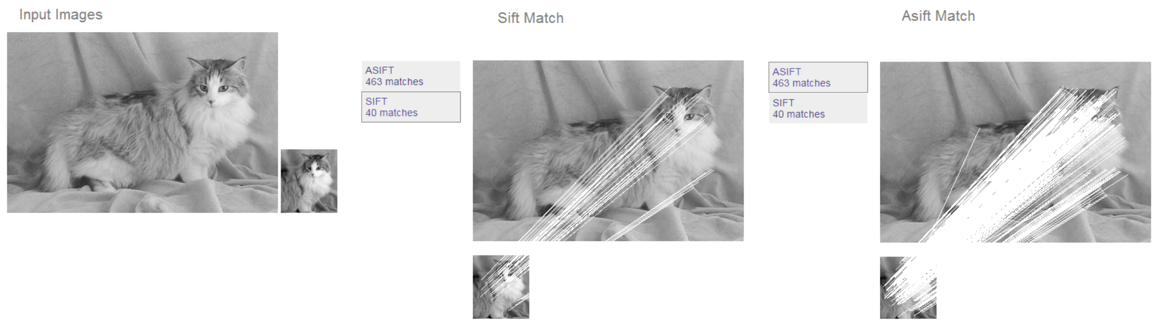


Figure 5.2: CV Attack with SIFT and ASIFT IMAGE: ©Ecole Polytechnique

trained and tested their algorithm of choice (e.g., SIFT) on various images of cats gathered from around the web and can recognize them with a good degree of accuracy. When he feeds the CAPTCHA challenge image into the algorithm, it returns a high probability of the image being of a cat. Using the noise generation algorithms, the image of the cat can be altered enough so that the CV algorithms return a low probability or cannot determine what the image is depicting, but a human can still determine it is showing a cat. The intention is to use the noise to distort the edges of scenes/objects and alter the patterns within the image enough such that various commonly used CV techniques fail to provide meaningful results for an attacker. Also image filters can be used to distort and move pixel neighborhoods such that detection and mapping algorithms fail to achieve matches and/or detect similarity. Figure 5.2 shows that both SIFT and ASIFT can overcome scaling issues (mappings are found to a smaller, cropped image of the cat), and ASIFT typically provides more mappings than SIFT.

### 5.1.2 Defensive Design Methodology

Our method is designed to work with existing image CAPTCHAs that rely on a database of images for challenges. After application of SIGNAC (our noise addition process), we demonstrate that the same database of images provides better security against RIS and CV based attacks. The MATLAB image processing toolbox is used to generate the new secure images. The function imnoise is used to add noise to the images. The test image set contains 100 images in total, 10 images in each of 10 different categories: airplane, bird, car, cat, doll, fish, flower, monkey, robot, and train. The categories are deliberately made "concrete" instead of abstract, as this makes it easier to create naming and distinguishing image CAPTCHAs that will be straightforward for user/usability testing. This also provides the CV algorithms with an "object" to recognize. The noise functions utilized in the method are the four generalized noise functions available in the MATLAB IPT[3].[36].

**Noise Functions** The noise functions utilized in the method are the four generalized noise functions available in the MATLAB IPT. A brief explanation of how each noise function operates has been included from the IPT reference manual[36].

*Gaussian* adds Gaussian white noise of mean m and variance v to the image I.

*Speckle* adds multiplicative noise to the image I, using the equation J = I+n*I, where n is uniformly distributed random noise with mean 0 and variance v.

The mean and variance parameters for 'gaussian' and 'speckle' noise types are always specified as if the image were of class double in the range [0, 1]. If the input image is of class uint8 or uint16, the imnoise function converts the image to double, adds noise according to the specified type and parameters, and then converts the noisy image back to the same class as the input.

*Salt & Pepper* adds salt and pepper noise to the image I, where d is the noise density. This affects approximately d*numel(I) pixels.

*Poisson* generates Poisson noise from the data instead of adding artificial noise to the data. If I is uint8 or uint16, then input pixel values are used directly without scaling. For example, if a pixel in a uint8 input has the value 10, then the corresponding output pixel will be generated from a Poisson distribution with mean 10.

While noising alone is sufficient to meet the criteria of security, frequently it

[3]http://www.mathworks.com/help/images/ref/imnoise.html

requires an amount of noise that is near or exceeds the limits for a human to be able to comprehend what is depicted in the image. To improve usability, image filters can be applied to provide additional alterations, lowering the amount of noise required to meet the security criteria. To this end we use the “motion” filter available in the fspecial function in the IPT. It returns a filter to approximate, once convolved with an image, the linear motion of a camera by len pixels, with an angle of theta degrees in a counterclockwise direction. The filter becomes a vector for horizontal and vertical motions.

**The SIGNAC Approach - System Design & Architecture** As discussed above, SIGNAC is implemented using the MATLAB Image Processing Toolbox. The script below gives an idea of the method in action. *X* is the image at the initial starting point when it is read into the IPT. The images are usually in the .gif, .jpg, or .png formats. When imported to MATLAB, they are converted to the type uint8 before noise operations are performed. *c1* through *c5* represent the image at various stages of its alteration. Note that this example is a multimethod output, as different noise and filter functions are being used to generate an image at each step. It is important to note that ordinality plays a large factor in the outcome of the image's success or failure in defeating an RIS engine, and will be discussed in the following section. This script is designed to create the image filter, read in the image file, apply noise, filter the image, then apply noise 3 more times before writing the image to a file.

```
f=fspecial('motion',11,3)
x=imread('1.jpg')
c1=imnoise(x,'salt & pepper',0.35)
c2=imfilter(c1,f)
c3=imnoise(c2,'speckle',0.35)
```

```
c4=imnoise(c3,'gaussian',0,0.35)
c5=imnoise(c4,'poisson')
imwrite(c5,'1', 'jpg');
```

This script represents the final script used to create the secure image set used in our experiments. Salt and pepper noise can be considered the most destructive type of noise, as it is the most extreme - changing pixel values to 0's and 1's. The motion filter is then applied to the image with a len of 11 pixels and a theta of 3 degrees counterclockwise, which serves to relocate the pixels that were changed with the addition of the salt and pepper noise to new areas around the image. This aids in obfuscation of clues about pixel values in a particular neighborhood, i.e., multiple pixels will now be distorted with values that differ from the original. After the filter is applied, multiplicative noise in the form of the speckle noise function distributes its noise in a uniform fashion throughout the image, followed by the addition of white Gaussian noise. The final step involves using the Poisson noise function, which does not add artificial noise, instead it generates noise from the image data and then applies it to the image using a Poisson distribution. This serves to further obfuscate the artificial noise that was added during previous steps by shifting the pixel values around.

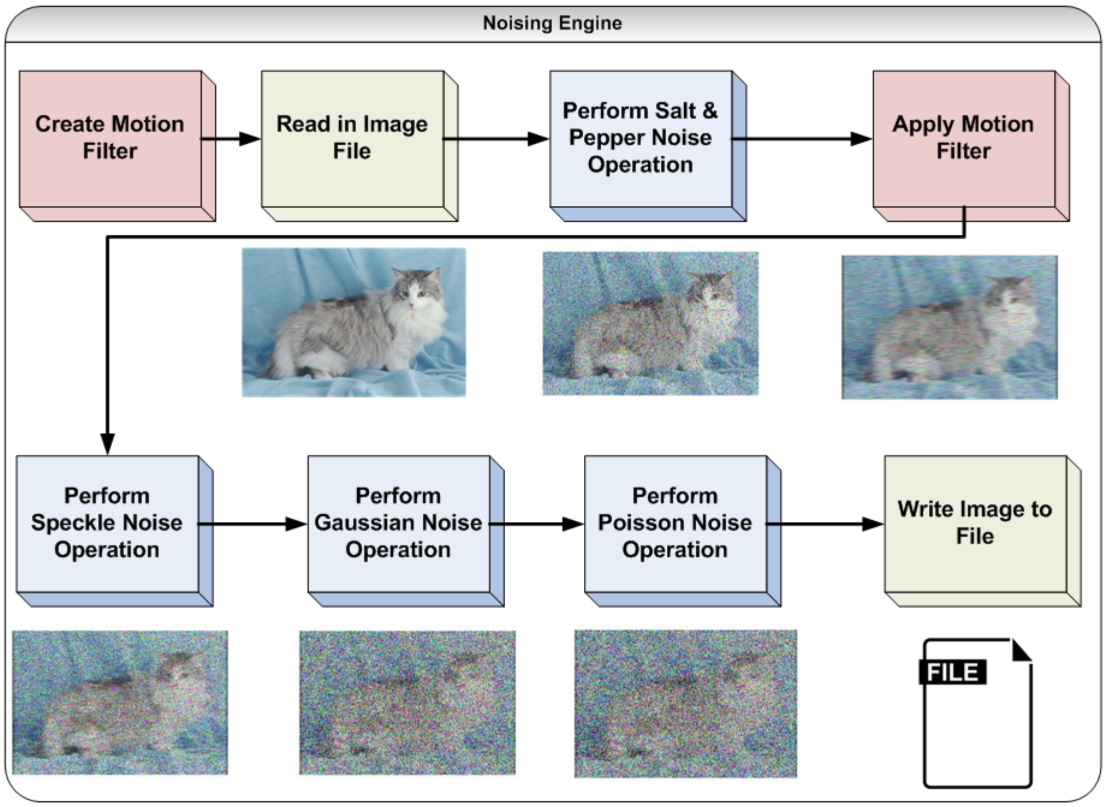

Noising Engine
Create Motion Filter
Read in Image File
Perform Salt & Pepper Noise Operation
Apply Motion Filter
Perform Speckle Noise Operation
Perform Gaussian Noise Operation
Perform Poisson Noise Operation
Write Image to File
FILE

Figure 5.3: Procedure for producing a noised image IMAGE: ©CIMIC Lab - Rutgers University

One interesting aspect of using noise generation algorithms to secure images is that the images produced by the algorithms we selected are very "grainy" or "pixelated" in appearance – very similar to a snowy TV picture. The noise introduced is primarily additive and multiplicative in nature, thus it tends to shift around color values in various pixels based on a threshold of our choosing. The benefit to this noise is demonstrated when the image is viewed as a matrix of numbers (as a computer would "see" the image), the values vary wildly and do not follow the patterns typical of a structured image. However, when viewed by a human eye (along with a human mind behind it), the colors blend into an image that is coherent and cognizable (the "Pointillism effect"). Strangely, this side effect of enhancing security actually does not impact usability negatively (to a point). In general, this effect is easier to achieve the further away your eye is from the image, or if the image is small in dimensions (scaled down). Figure 5.3 demonstrates this process in action. Note that this example is a multimethod output, as different noise and filter functions are being used to generate an image at each step. It is important to note that ordinality plays a large factor in the outcome of the image's success or failure in defeating an RIS engine.

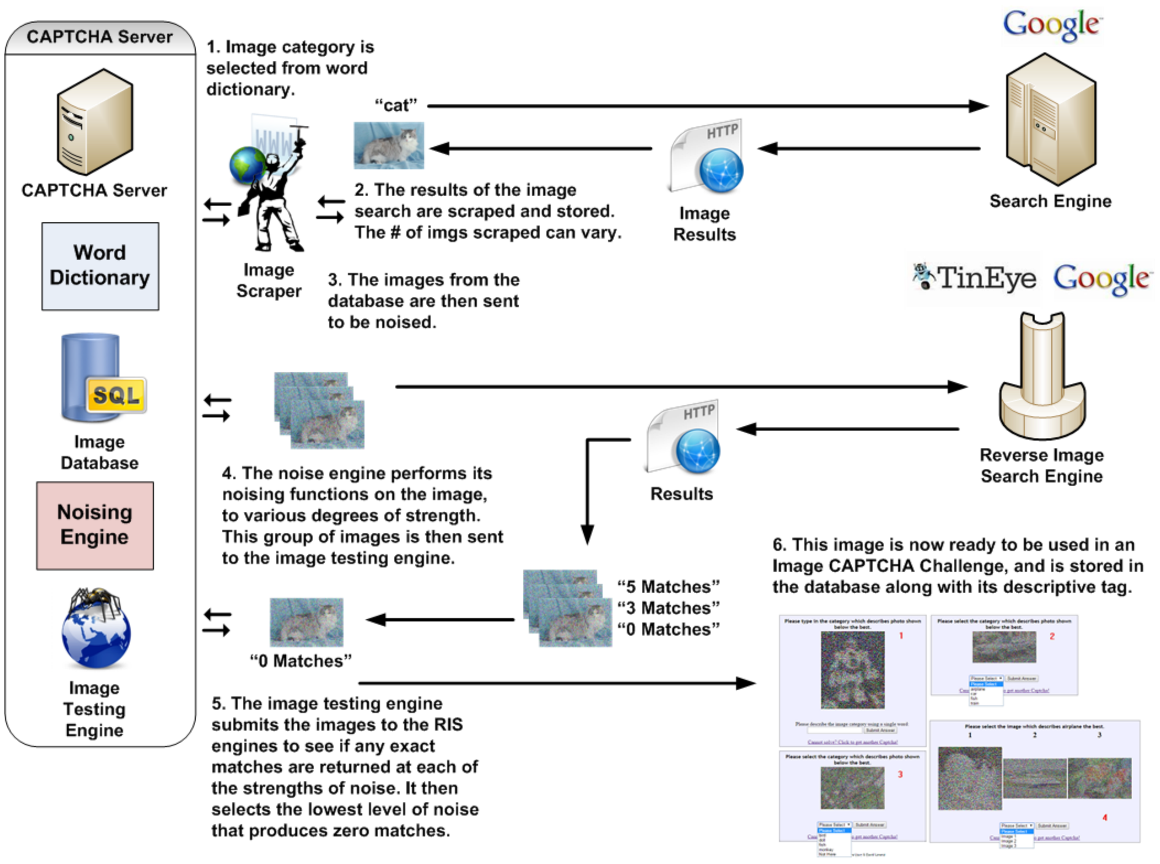

CAPTCHA Server
CAPTCHA Server
Word Dictionary
Image Database
Noising Engine
Image Testing Engine
1. Image category is selected from word dictionary.
"cat"
Google
Search Engine
Image Results
Image Scraper
2. The results of the image search are scraped and stored. The # of imgs scraped can vary.
3. The images from the database are then sent to be noised.
TinEye Google
Reverse Image Search Engine
Results
4. The noise engine performs its noising functions on the image, to various degrees of strength. This group of images is then sent to the image testing engine.
"5 Matches"
"3 Matches"
"0 Matches"
"0 Matches"
5. The image testing engine submits the images to the RIS engines to see if any exact matches are returned at each of the strengths of noise. It then selects the lowest level of noise that produces zero matches.
6. This image is now ready to be used in an Image CAPTCHA Challenge, and is stored in the database along with its descriptive tag.

Figure 5.4: Image generation for CAPTCHA challenge IMAGE: ©CIMIC Lab - Rutgers University

One of the more challenging aspects to generating secure images is that the level of noise required to prevent the RIS engine from returning a match varies with each individual image. As such, we have devised a system to handle the testing of images created with the noising process to ensure that any image used in a challenge will return zero RIS matches. The image testing engine provides feedback to the noising engine when generating the challenge images by submitting the noised image to an RIS engine and seeing if any matches are returned. The noising engine generates images in a stepwise manner (starting from 0.10 mean noise), incrementing the mean noise in the image by +0.05 each time until zero matches are achieved or a value of 0.45 mean noise is reached (the image is discarded if it goes higher, as humans will be unable to understand it). If desired, the image testing engine can have the noising engine stepwise decrements the mean noise by -0.01 until a match is returned again and then increase that value by +0.01 - providing the finest granularity for the minimum amount of noise required to defeat the RIS engine for that image. This gives the human user the best chance at comprehending the image while still meeting the security requirements. Figure 5.4 demonstrates such a system in action.

**SIGNAC Demonstration Scenario** For our live demonstration scenario, we provide an example security and usability analysis of an image found online using an image search engine with our image challenge testing script (this script is analogous to the image testing engine described in the previous system architecture section). This script is designed to interact with MATLAB and Tineye to test prospective images to be used as challenges in one of the four CAPTCHA schemes. The script is designed to provide a quick visual overview of the incrementalism required in testing various images against the "zero matches" security requirement to stop reverse image search attacks and hinder computer vision attacks. One benefit to having a script such as this is that it allows image CAPTCHA researchers to test and look for features, styles, and composition in various images that lend themselves better for security use. This method of presentation also makes it easier to have a mechanical turk or other human method of verification provide answers to the "usability" of a particular noised image.

**RIS Engine Probing** Figure 5.6 shows an example of a single image test working against the RIS engine Tineye. For the original figure (5.6a), Tineye provides exact match results.

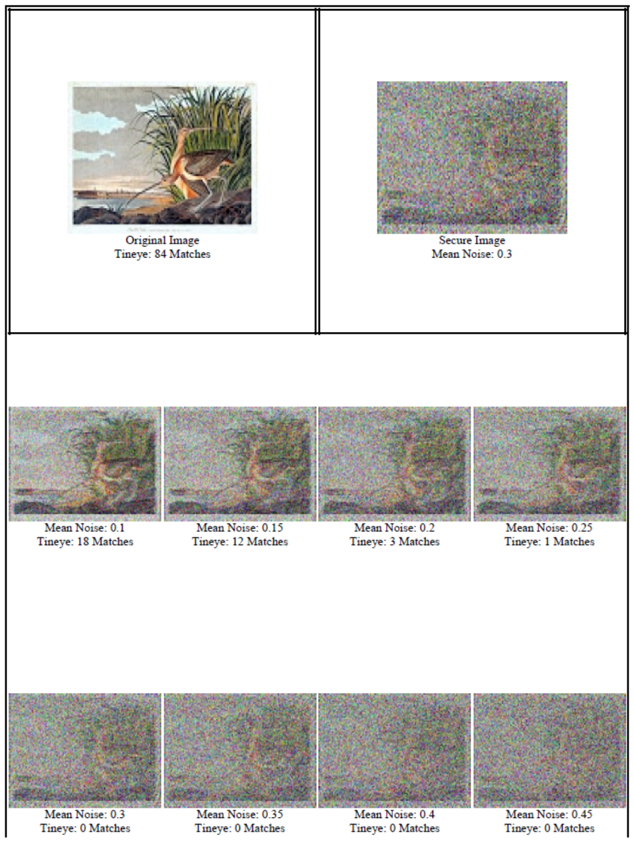
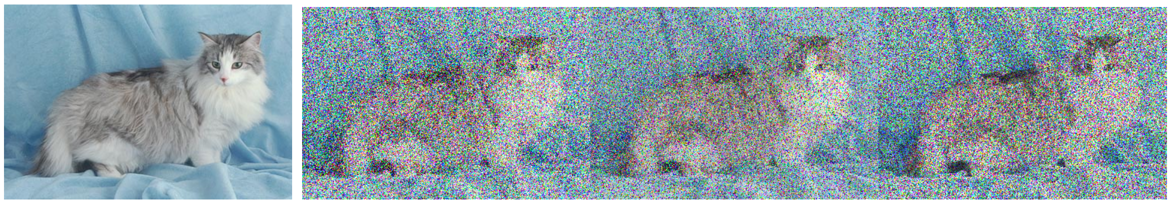


Figure 5.5: Image Analysis for CAPTCHA challenge IMAGE: ©CIMIC Lab - Rutgers University

**Single Noise Function, Single Stage** Currently, the initial image returns 16 exact matches from across the web. These results were gathered using a single image noise function in a single step on the original image to produce an image that returns 0 exact matches. Note that these values are unique to this image, and vary based on the image properties.



(a) RIS Probe Test Image (b) Gaussian Noise with 0.2 Mean (c) Salt & Pepper Noise with 0.3 Mean (d) Speckle Noise with 0.4 Mean

Figure 5.6: RIS Engine Probing IMAGE: cat ©Google Image Search

**Ordinality Test** This test demonstrates the ordinality of noise functions. These tests were run with the default settings of each noise generation function to see if the order in which the functions are applied affects the results. Table 5.1 gives the results with the different orders. Note that these results were obtained using the original cat image, and these results apply only to that image.

Table 5.1: Results of Ordinality Test

| Primary Noise Function | Permutations | # of Results | Primary Noise Function | Permutations | # of Results |
|---|---|---|---|---|---|
| Gaussian (G) | GKPS | 10 | Speckle (K) | KGPS | 09 |
| | GKSP | 10 | | KGSP | 11 |
| | GPKS | 11 | | KPGS | 10 |
| | GPSK | 10 | | KPSG | 10 |
| | GSKP | 10 | | KSGP | 11 |
| | GSPK | 10 | | KSPG | 11 |
| Salt & Pepper (S) | SGKP | 11 | Poisson (P) | PGKS | 10 |
| | SGPK | 11 | | PGSK | 09 |
| | SKGP | 09 | | PKGS | 10 |
| | SKPG | 10 | | PKSG | 10 |
| | SPGK | 11 | | PSGK | 10 |
| | SPKG | 09 | | PSKG | 09 |

From these results, it is clear that order makes a difference as the range for matches is +-2 matches, with a high of 11 matches and a low of 9 matches. We then further investigate the chain of methods that produce the least amount of matches.

```
x=imread('cat.jpg')
c1=imnoise(x,'salt & pepper',0.11)
c2=imnoise(c1,'poisson')
c3=imnoise(c2,'speckle',0.11)
c4=imnoise(c3,'gaussian',0,0.11)
imshow(c4)
```
(a) Minimal Equal Noise via (SPKG)

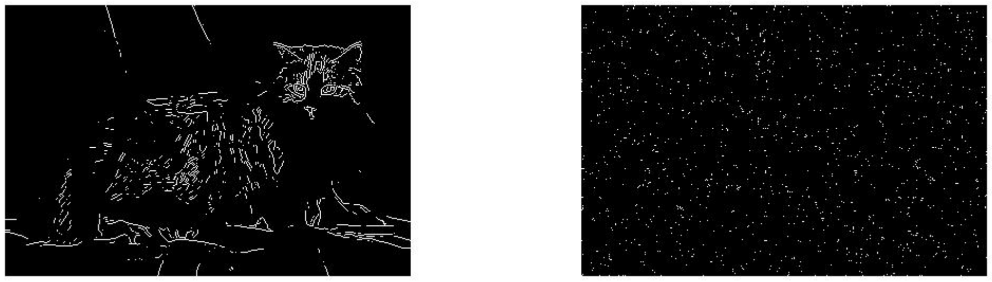
```
x=imread('cat.jpg')
c1=imnoise(x,'salt & pepper',0.11)
c2=imnoise(c1,'poisson')
c3=imnoise(c2,'speckle',0.05)
c4=imnoise(c3,'gaussian',0,0.05)
imshow(c4)
```
(b) SPKG Current Working Minimum

Figure 5.7: SIGNAC MATLAB Scripts

**Threshold Determination** After deciding on an appropriate ordinality for noise methods, it must be determined the minimum threshold at which zero matches are reached - the key to our anti-RIS security criterion. A rough metric is used first, incrementing each mean value by 0.1 until zero matches are found. Then it decrements by 0.05 until a match and then increments or decrements by 0.01 until the minimal value is reached with zero matches. Figure 5.7 shows two scripts that embody these principles in action. Note that both scripts provide zero matches, however, the second script produces a clearer image because less noise overall is added during the application of additional functions. This is important for usability reasons - as the clearer the image is, the easier the chance a real human will have in recognizing what it depicts.

However, the values are extremely sensitive. For example, decrementing the mean in the initial noise function of the previous script by 0.01 to 0.1 produced 5 matches. Decrementing both means c3 and c4 by 0.01 each produced 8 matches. It is a painstaking and involved process to tune each image for a working minimum. Unfortunately, this process must be done for each image on an individual basis and cannot be generalized beyond offering a rough threshold for which any series will return zero matches, and this threshold is usually quite high and may impact usability.

As such, this script serves as the endpoint for security against RIS engines, as zero matches are returned with these values. Computer vision based attacks are an entirely different subject, and there are no guarantees that this RIS minimum will



(a) Original Image Edge Detection (b) Edge Detection after Noising

Figure 5.8: Edge detection tests IMAGE: ©CIMIC Lab - Rutgers University

have any impact on the ability of CV tools to perform recognition tasks.

**Noise for Anti-Computer Vision** While stopping RIS engines was the primary challenge, CV tools are powerful and have been successfully used to defeat image based CAPTCHAs in the past. Thus, we aim to make it as difficult as possible to use them in performing object recognition tasks. One such CV attack case is that of edge detection. This is a key component of object recognition, and being able to foil it will go a long way in stopping any CV attacks from performing this task on an image recognition CAPTCHA challenge.

In Figure 5.8a, we can see the results of a Sobel edge detection run on the test image. It clearly depicts a cat, while also picking up some of the wrinkles in the sheet behind the cat. Enough detail of the cat comes through that a CV algorithm could make a decision about what is depicted in the image. Note that when edge detection is performed, the image is first converted from RGB to grayscale, and then to binary (hence the black & white) after the edge detection algorithm is run. Figure 5.8b shows the same Sobel edge detection method run on the image of the cat that has been noised. It can be seen that the cat has completely disappeared - only white dots on a black background appear. No useful information can be gained from this image.

In Figure 5.9, we can see that when we compare a clear image to the noised image, it does not return any keypoint matches. Note that the noise significantly increases the number of detected keypoints (more than double the clear image), suggesting that the noise is effective in throwing off attempts to match the two images.

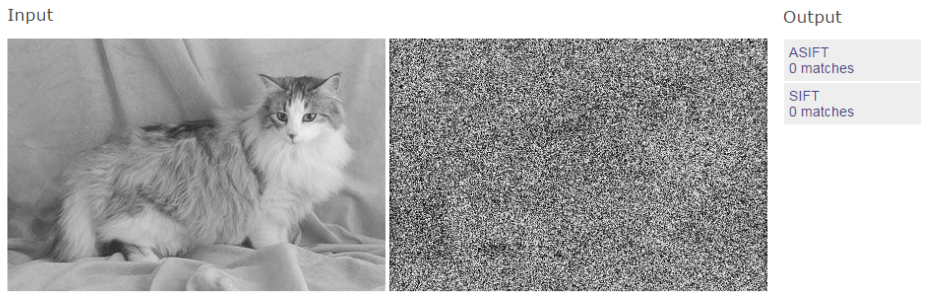


Figure 5.9: SIFT & ASIFT image matching IMAGE: ©Ecole Polytechnique

### 5.1.3 Experimental Results and Analysis

We now describe the experimental evaluation to test image security against both RIS engine attacks and CV attacks using the aforementioned online tools. We have gathered 100 random indexed images from 10 categories and applied the method described in Section 4. Note that the image filter values did not change during the course of the experiments, only the mean values of the noise functions.

**RIS Engine Testing** The goal of this experiment was to establish a baseline for which a set of noise functions can provide zero exact matches against both Google (G) and Tineye's (T) reverse image search engines. As mentioned in the Methodology section, the approach we use is more conservative from the security perspective, in that many of the images are no longer returning matches at much lower levels of noise overall. We consider even 1 match a failure - thus we do not report specific numbers of matches for each image failure. The number following the search engine designation is the number in the set of 10 for that category, e.g., car contains 10 images total,

Table 5.2: RIS Engine Testing

| CategoryID | Category | Noise Functions at 0.25 Mean | | Noise Functions at 0.30 Mean | |
|---|---|---|---|---|---|
| | | Pass | Fail | Pass | Fail |
| 1 | airplane | T,G | | T,G | |
| 2 | bird | T,G | | T,G | |
| 3 | car | G | T22 | T,G | |
| 4 | cat | T | G32 | T,G | |
| 5 | doll | T,G | | T,G | |
| 6 | fish | T | G57 | T,G | |
| 7 | flower | T,G | | T,G | |
| 8 | monkey | T | G77,G79 | T,G | |
| 9 | robot | | T84,T85,G81,G84,G85 | | T84,G84 |
| 10 | train | T,G | | T,G | |

numbered 21-30 - T22 means that image 22 failed to produce zero matches as matches were found on Tineye (but not Google).

Table 5.2 shows that at 0.25 mean noise, we have 8 out of 100 unique images returning matches. Tineye has one unique hit (T22) and Google has five unique hits (G32,G57,G77,G79,G81). There is overlap on image 84 and 85 as both engines returned matches. This means that out of our random sample of 100 indexed images, 92 out of 100 returned zero matches. At 0.30 mean noise, we have 1 out of 100 unique images returning a match. Tineye and Google both return matches for the same image. This means that out of our random sample of 100 indexed images, 99 out of 100 returned zero matches. At 0.35 mean noise, we have achieved our goal of returning zero matches for our test group of 100 random indexed images.

**Computer Vision Testing** In this section, we evaluate the effectiveness of SIFT and ASIFT to provide object detection and image matching. The key takeaway is to fool the keypoints calculator into examining incorrect correspondences by inflating the number of keypoints in an image or not finding any matches due to an insignificant matching value. Figure 5.9 demonstrates failure to find matching keypoints on an exact image match (original clear image vs. noised image).

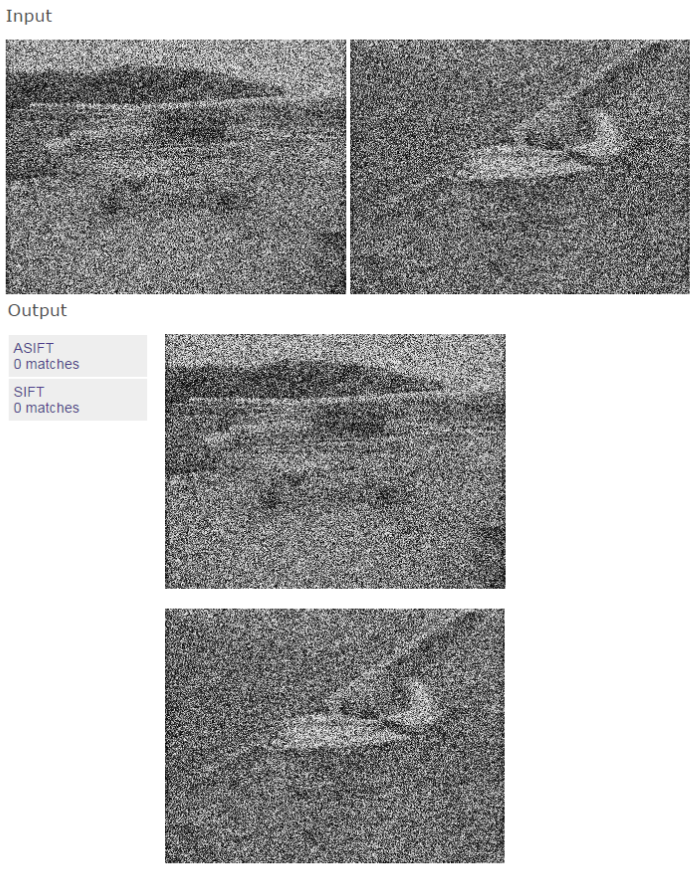


Figure 5.10: Inter-Category Test: Airplanes of a similar build noised IMAGE: ©Ecole Polytechnique

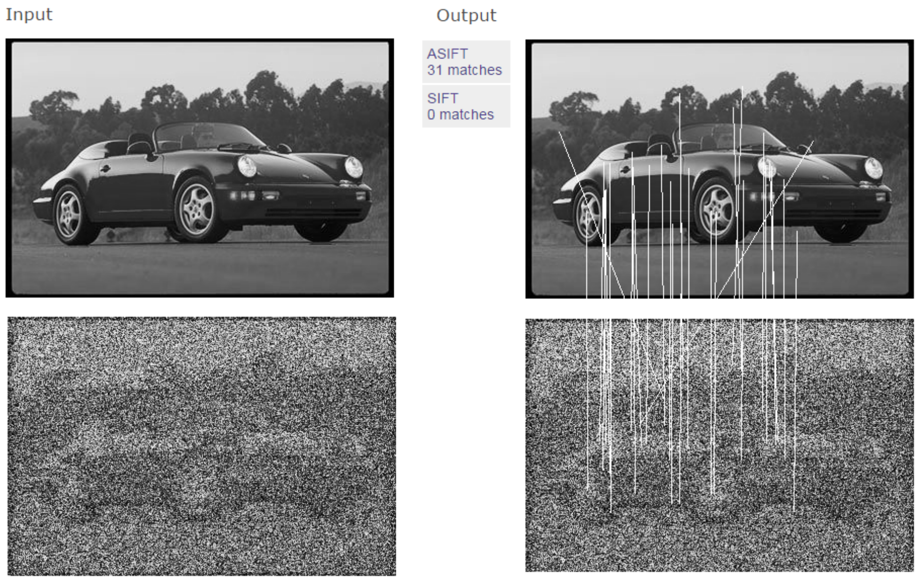


Figure 5.11: Exact Match Test: Original Vs. Noise with false positives IMAGE: ©Ecole Polytechnique

It may be observed from Figure 5.11 that SIFT has returned zero matches, but ASIFT has returned 31 matches. However, upon further investigation, we can see that some of the matches are false positives. More specifically, we can see that some of the points provided are incorrect, as it seems the noise has been mistaken for keypoints. However, in both cases, SIFT has returned zero matches, and caused enough doubt in the ASIFT responses, thus discouraging potential attackers.

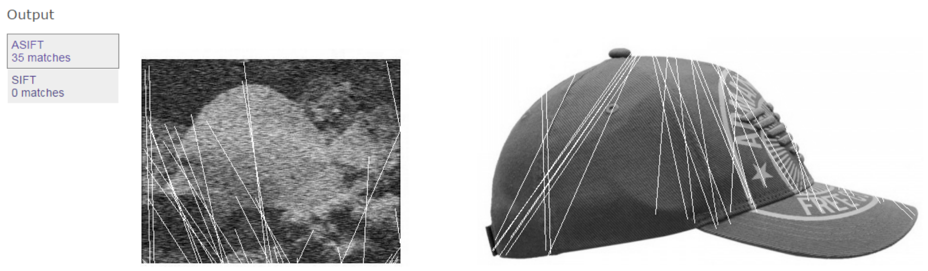


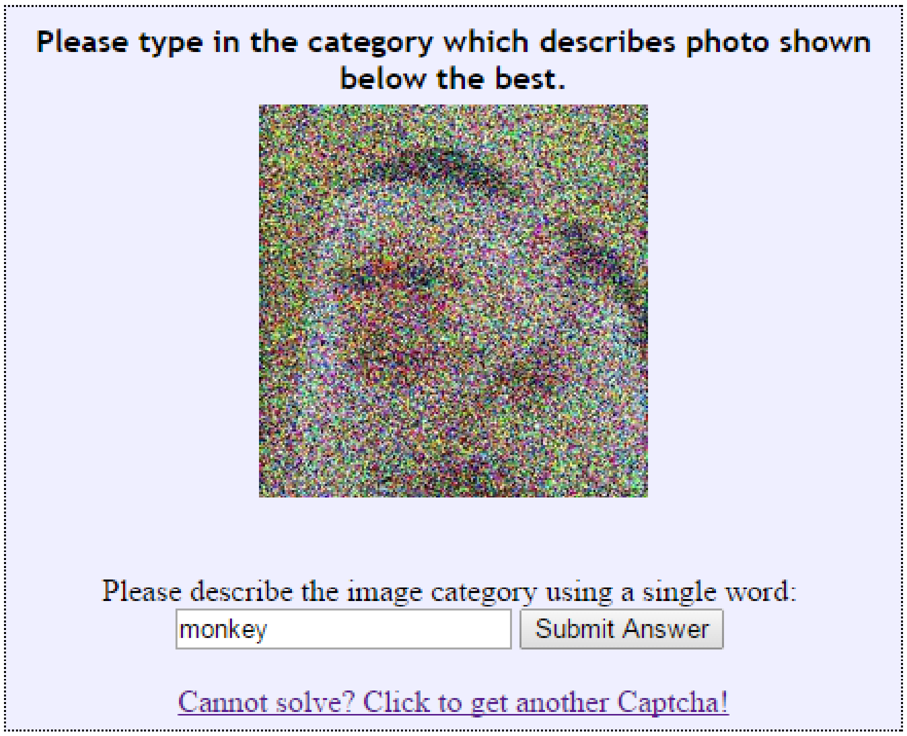
Figure 5.12: Shape Test: Noised image with similar shape image IMAGE: ©Ecole Polytechnique

In Figure 5.12, we can see that ASIFT was fooled by a similarly shaped image. In this case the fish and the hat have a similar shape, and it was enough to return matches, even though clearly the two images are quite different. It is worth noting that in this example, as we scaled the size of the hat image, the ASIFT results dropped to zero. Note that the ASIFT and SIFT engines are sensitive to slight changes in any images (noised or otherwise), and thus generalizing the results to all images will require more study.

### 5.1.4 Experimental CAPTCHA Style Implementations using SIGNAC

A design perk of using noise to obfuscate an image is that it makes image fingerprinting a significantly more challenging task. Each time the algorithm outputs an image, since the noise is randomly generated and applied to the image in multiple layers, each image comes out, for all intents and purposes, as unique. In truth, the sample space complexity for possibilities is quite large, as it is the size of the image in pixels times the color depth in the RGB channels. Note that this sample space complexity changes with each image. Simple hashing comparisons demonstrate that the same algorithmic process outputs enough variety in images via randomness in the noise methods that the same base tagged image can be used in multiple challenges as long as its run through the noising engine each time a challenge is generated. The second beneficial outcome of using noise is that the image produced after the application of the noise algorithms is essentially a “new image”, meaning that when comparison methods for reverse image search are used, there is enough of a delta between the original image and the noised image that no matches will be returned by the search engine. The third beneficial outcome of using this method is that it helps to obfuscate patterns and features depicted in the image by introducing noise throughout the image. This helps to stop methods like edge detection methods and scale invariant feature transformations for aiding attackers in determining features about what is depicted in the image scene.

An in-depth discussion of the design choices for our experimental image based CAPTCHAs follows. The goal is to provide an overview of different options available within a single style, such as image based CAPTCHAs. To serve as the tagged image database, 100 noised images were generated in total, 10 images gathered from a web search in each of 10 different categories. The 10 image categories were chosen to be “concrete”, to lower ambiguity for the user (airplane(1), bird(2), car(3), cat(4), doll(5), fish(6), flower(7), monkey(8), robot(9), train(10)). All 5 styles have the option to click a link to serve up a new CAPTCHA if the user cannot understand/decipher/solve the one they have been given. This is tallied as “no response” by our database. To test the usability of the noise method on human users, we designed 5 different styles of image CAPTCHA with varying degrees of difficulty. Style 1 displays an image and asks the user to describe it by entering a description (freeform text response). Style 2 displays an image and provides a dropdown box with 4 responses, with 1 of the 4 choices being the correct answer. Style 3 displays a dropdown box with 5 responses, 4 choices and not here. Style 4 asks the user to select the image from 3 images that best represents X, where X is an image category. Style 5 uses a sentence with the "Cambridge effect" applied to it to provide contextual clues for the missing keyword that would complete the sentence that is related to an image depicted in the CAPTCHA challenge. In our opinion, the order of difficulty is (1,5,3,2,4) from most difficult to least difficult. One final security measure included in our challenges is that the default response is always "Please Select", which evaluates to an incorrect response if a bot simply tries to guess the first available response and submit it over and over to pass the challenge. A time-to-live is also available (e.g. please solve the challenge within 30 seconds or another challenge will be presented) as an option.


**Please type in the category which describes photo shown below the best.**

Please describe the image category using a single word:

monkey

Figure 5.13: Style #1: Freeform Text Response IMAGE: ©CIMIC Lab - Rutgers University

Figure 5.13 demonstrates an example of an image based CAPTCHA with little to no clues provided by the challenge. The only clue is that the challenge is a classification task, based on the instructions asking the user to provide a descriptive category using a single word in a text box. SIGNAC provides a noised image so that attackers cannot easily use a reverse image search engine and must expend extra effort on specialized computer vision algorithms to attack it. This makes it more difficult for an attacker to use clever tricks to circumvent the task, as there is minimal information disclosure in its challenge presentation (i.e. not multiple choice). However, from a user perspective, this particular presentation of the style is perhaps the most challenging, due to the fact that the user must guess what we are asking for as the category of the solution to the challenge. When designing a freeform text response, effort must be expended on behalf of the designer to provide reasonable accommodations to the correct solution. For example, the image shown in 5.13 could reasonably have the following category labels: car, automobile, auto, Porsche, sports car etc. If we are only accepting car as the correct response, the user would most likely get this wrong and become frustrated if they offered one of the other answers. Case sensitivity and spelling also present problems that must be addressed by the designer using freeform text response if decent usability is to be achieved. This is a prime example in the trade off between secure design and usability.

Figure 5.14 demonstrates our easiest implementation of the image based CAPTCHA. A dropdown box with four possible options available - one of which is the correct answer. The dropdown box in this implementation is populated at random by categories from the image database, along with the correct response that correlated to the challenge image. An attacker can exploit this over time to gather all of the categories used in the image database and increase their chances of a successful attack. While the classification task remains the same as in 5.13, the contextual clues provided by the dropdown box help narrow the scope of the classification task down for the user.

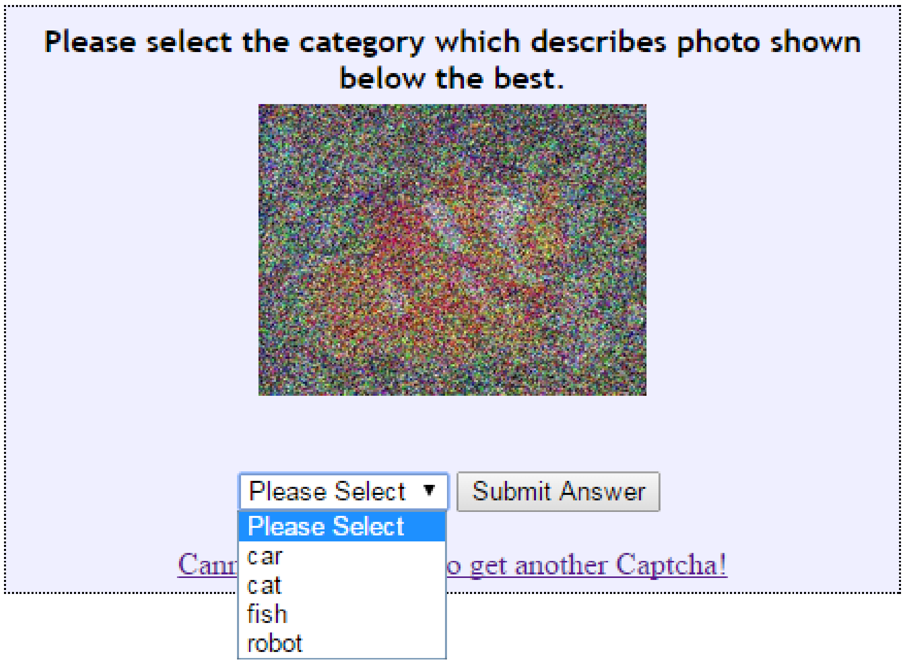


Figure 5.14: Style #2: Four Choice Dropdown IMAGE: ©CIMIC Lab - Rutgers University

If the user perhaps cannot identify the noised image just by looking at it alone, seeing what choices are available can aid in answering the challenge correctly. The choice of the number of options has a direct effect on security – the number of options is directly correlated with the chance of a successful guess. 3 choices (33.3%), 4 choices(25%), and 5 choices(20%) are all options we explore. One option to consider is to populate the list with a large number of choices to reduce the chance of a successful guess, perhaps even having a list of words that are never used as image category just to confuse an attacker. However, this most likely will also have a negative impact on usability, as now people are forced to spend more time on the challenge considering all of the options, and perhaps incorrectly choosing from the categories that are not used.

Figure 5.15 demonstrates the addition of a unique option to style # 2 - the choice of "Not Here". "Not here" can have several interesting applications – for example it can be used in conjunction with a weighting algorithm to provide a bonus points

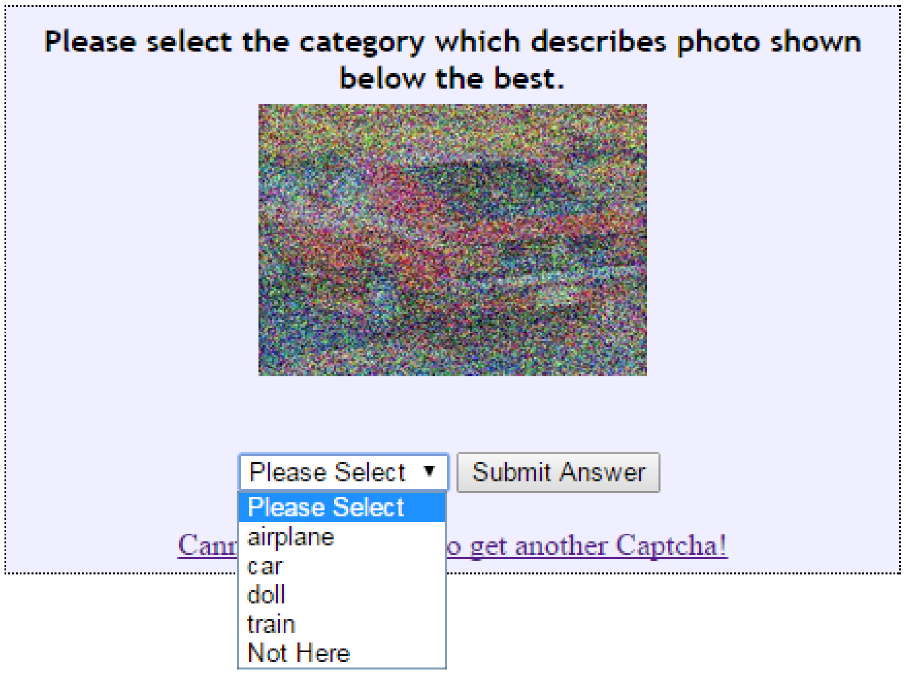


Figure 5.15: Style #3: Five Choice Dropdown w/Not Here IMAGE: ©CIMIC Lab - Rutgers University

to a scoring algorithm that determine series performance over time. Imagine asking a user to solve a series of challenges in this style, where progressive correct answers yield a higher score of humanity. A correctly identified "not here" can be weighted higher after a series of correct answers from the user,perhaps allowing them access to the service or form without having to solve more challenges. However if "not here" is not correctly identified, the penalty can be increased to a level higher than a regular incorrect response and force more challenges to be provided before access is granted. The intention of "not here" as an answer choice is to force the challenge evaluater to guess or perform the categorization task, which can then be dealt with accordingly.

Figure 5.16 demonstrates the use of a keyword in the instructions for the solution. In this case, it asks the user to identify which of 3 images best describes "bird". While this provides an attacker with the correct category, they must perform the the categorization task of evaluating three images. The images are labeled in sequential order from left to right (1,2,3). Users have an easier time with this style, as the

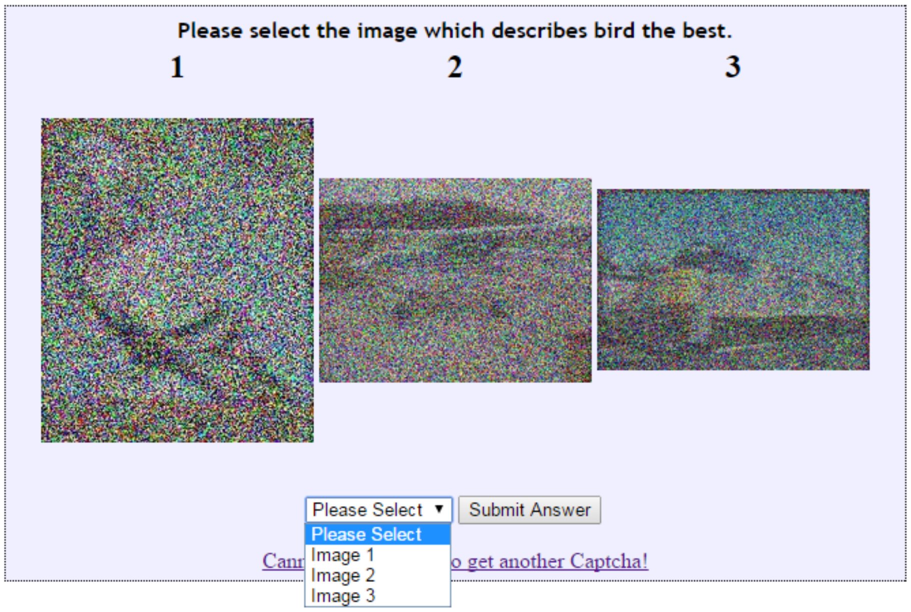
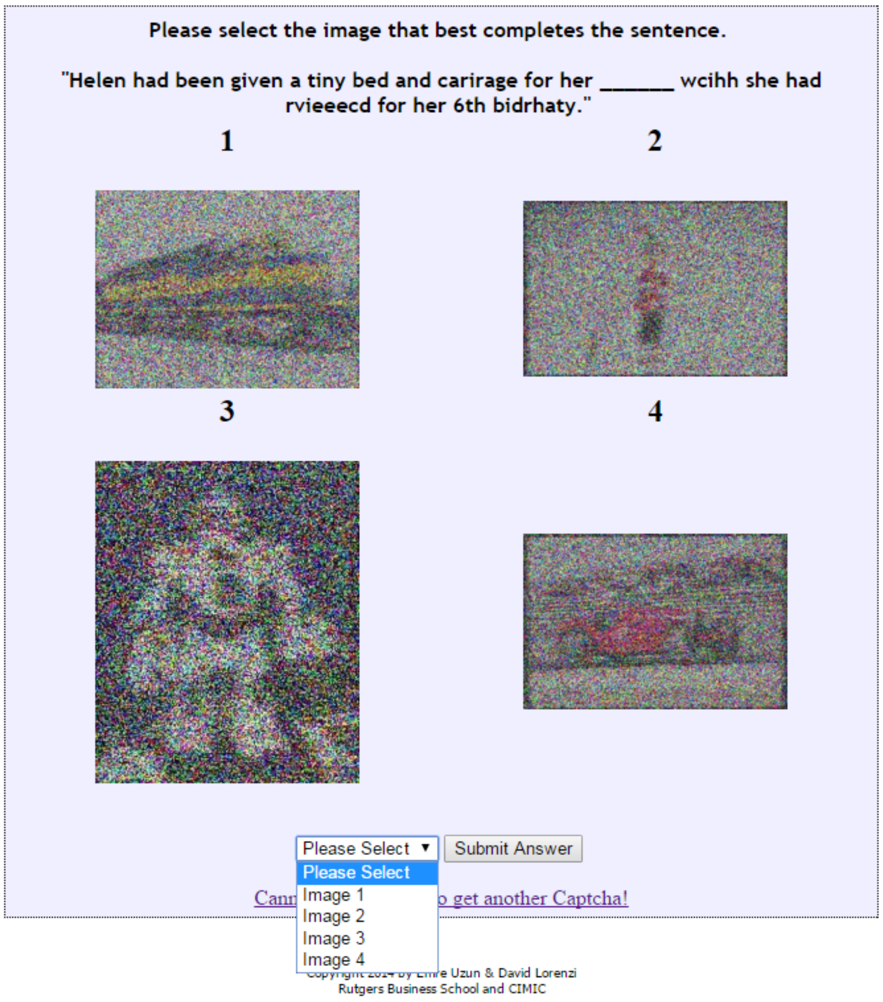


Figure 5.16: Style #4: Select Image from three based on Keyword IMAGE: ©CIMIC Lab - Rutgers University

keyword allows them to evaluate the images according to their notions of what the keyword should depict. In this way, it guides the answer without disclosing too much information to help an attacker. It is worth noting that this method is scalable, in that you can display multiple images in a grid and ask for multiple correct responses (with the correct number changing each time you serve the challenge) for a particular image category. You can also have a parallel database of garbage images used to fill out the challenge image grid to make it harder for an attacker to build a database of challenges (SIGNAC also helps keep the correct solutions secure as well). This particular implementation is the most basic form of this idea. Note that Google new image based CAPTCHA uses a form similar to this style.

Figure 5.17 represents our latest and most advanced form of image based CAPTCHA. The goal of this style is to provide obfuscated contextual clues for the keyword in the form of a sentence that help a human gain an edge over a machine when attempting to solve the challenge. The extra complexity is designed to slow down a machine, however a human should have no problem getting through it. The final version of



Please select the image that best completes the sentence.

"Helen had been given a tiny bed and carirage for her _______ wcihh she had rvieeecd for her 6th bidrhaty."

Figure 5.17: Style #5: Select Image from four Sentence Based Contextual Clues IMAGE: ©CIMIC Lab - Rutgers University

this style would be an automated system that can use random sentences pulled from various sources and select keywords that match an image in a tagged image database and present it as a challenge while still maintaining enough context in the sentence that a user can complete it. This style leverages the "Cambridge effect" on a sentence with the keyword omitted to help obfuscate the keyword from attackers and uses the SIGNAC method to provide secure images.

### 5.1.5 Limitations

As with all noise generation functions, there exists the possibility that their alterations to the image can be significantly decreased with smoothing functions/image filters or reversed entirely by an appropriate function. Sufficiently advanced attackers with image processing experience may be able to reverse some of the distortion effects that come as a result of the noise generation to the degree that the image becomes vulnerable to RIS or CV attacks. We attempt to minimize this weakness by using randomness in the function when applying the algorithms to the images, as well as using image specific properties to provide alterations within the image. We believe the method has enough merit to be explored further and that the CAPTCHA security community will provide the appropriate level of vetting of our methodology in due time.

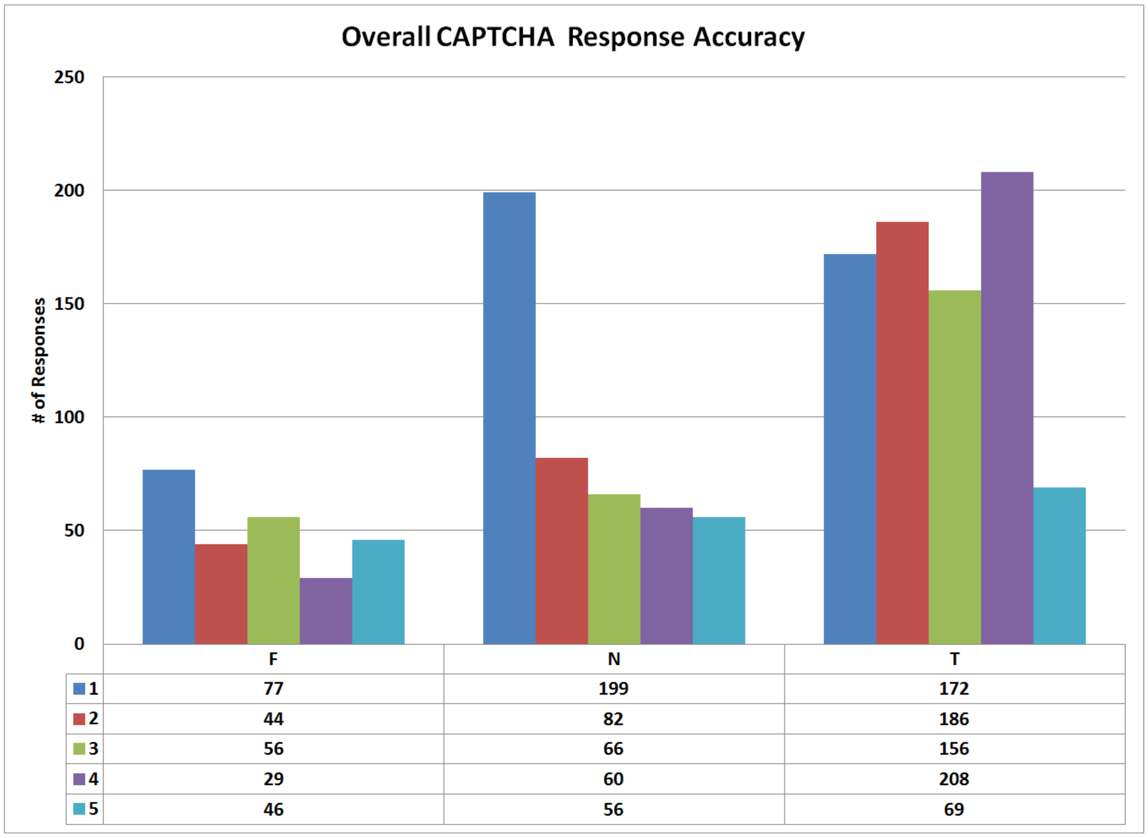
Secondly, there exist images that cannot be satisfactorily "noised", more specifically, the image will either fail to be recognizable by a human due to the excessively high level of noise added to the image to provide the security guarantee, or it will be recognizable to a human but fail to meet the security guarantee because the noise level is too low. This occurs when the image does not have colors (e.g. it is mostly composed of black and white.)

## 5.2 SIGNAC Testing & Usability

This section covers the usability study conducted to test SIGNAC usability on our series of basic image recognition CAPTCHAs.

### 5.2.1 SIGNAC Usability Study

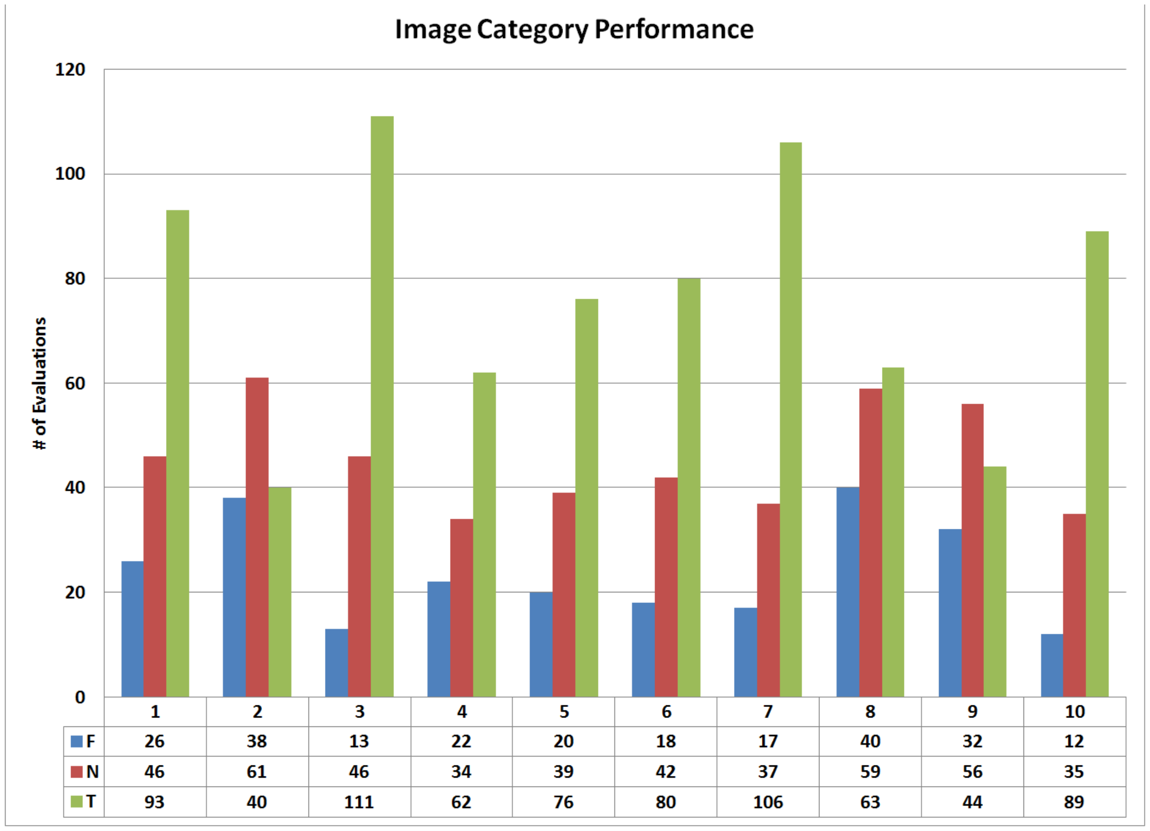
This user study is the culmination of our various lines of research into different aspects of CAPTCHA challenge design and security, specifically around issues relating to image based CAPTCHAs. This experiment serves as a way to gather real world usage data for our methods and designs. The user study aims to test both the usability of the SIGNAC method, as well as the usability of the design methodology of the five styles. We presented these challenges to approximately 100 undergraduate students in enrolled in courses across the business school. The challenge style and images used were selected at random by the CAPTCHA server when it generated the challenge.



| | F | N | T |
|---|---|---|---|
| 🟦 1 | 77 | 199 | 172 |
| 🟥 2 | 44 | 82 | 186 |
| 🟩 3 | 56 | 66 | 156 |
| 🟪 4 | 29 | 60 | 208 |
| 🟦 5 | 46 | 56 | 69 |

Figure 5.18: Overall CAPTCHA Response Accuracy - F = Incorrect N = No Response T = Correct

Figure 5.18 demonstrates the overall responses gathered from the students, divided into three groups. The F group is incorrect responses related to the challenge of each style. Not surprisingly, the freeform text response (style 1) performed the worst, with the highest number of incorrect answers. Style 4 performed the best with the lower number of incorrect answers. The N group represented when users could not solve the challenge so they clicked the link available at the bottom to get a new challenge – what we termed "No Response". Again, we see an extreme outlier in the freeform text response(style 1) - more people were unable to answer it than were able to answer it correctly. Somewhat interesting is that style 3, which is not significantly different in design from style 2 has a higher rate of incorrect responses simply by adding in "not here" and introducing ambiguity. Surprisingly, style 5 seems to be on a similar path to style 1, in that it has a large number of no repsonses and incorrect responses. We speculate this is because we do not provide an explanation of the "Cambridge effect" in the challenge, the user is simply presented with a scrambled sentence and must make sense of it.

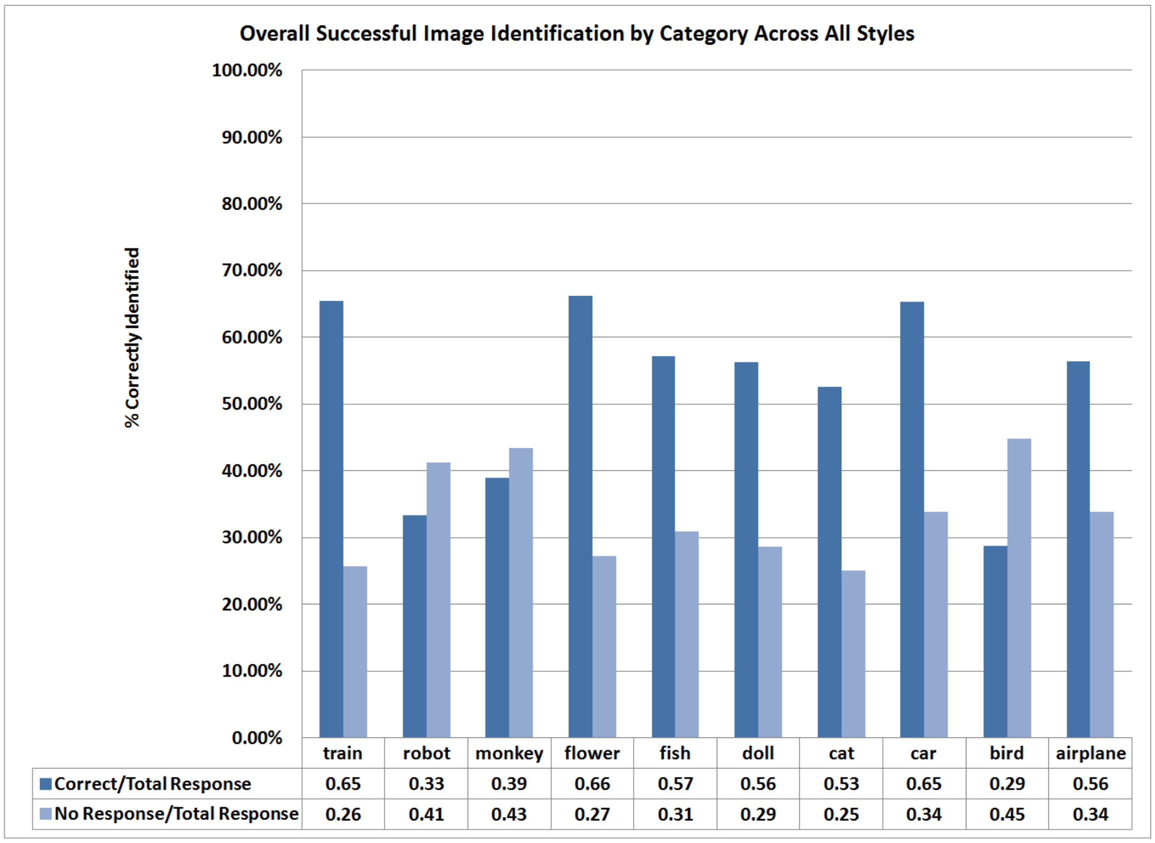


| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| 🟦 F | 26 | 38 | 13 | 22 | 20 | 18 | 17 | 40 | 32 | 12 |
| 🟥 N | 46 | 61 | 46 | 34 | 39 | 42 | 37 | 59 | 56 | 35 |
| 🟩 T | 93 | 40 | 111 | 62 | 76 | 80 | 106 | 63 | 44 | 89 |

Figure 5.19: Image Category Performance - F = Incorrect N = No Response T = Correct (airplane(1), bird(2), car(3), cat(4), doll(5), fish(6), flower(7), monkey(8), robot(9), train(10))

Figure 5.19 shows how well each image category did overall in the experiment. This helps us see which “concrete” categories work well and which ones perform poorly. Three categories that stand out as performing poorly are birds(2), monkeys(8), and robots(9). We speculate that this is due in part to many of the images chosen for this category depict birds and monkeys whose bodies are designed to blend into their natural environments. Thus, when the noising algorithm is applied, it can make it very difficult to discern the animal from the overall background image. The failure of the robots category is most likely due in part to the wide range of styles that a robot can come in - too much ambiguity. We can see that these categories had more no responses than they did correct or incorrect answers. Three categories that did well were cars(3), flowers(7), and trains(10). We suspect these categories did well because of the structured nature their designs - cars and trains do not change shape much, and flowers share similar shapes and come in bright and eye catching colors. Fish(6), doll(5), cat(4) and airplane(1) were in the middle of the pack with reasonable performance. Figure 5.20 shows the successful rate of identification of an image across all styles. Note that total response includes no response.

Finally in Figure 5.21 we provide a statistical overview of the performance of each style under different circumstances. We provide two scenarios for your contemplation - one where we have classified no response as a “correct” response and one where we have classified no response as an “incorrect” response. Imagine a scenario where a CAPTCHA is guarding a resource that is extremely valuable to the user - for example, their personal email account. Under these circumstances, the user will continue to answer CAPTCHAs until they can gain access to their user account due to the level of importance it has to them. It doesn’t matter if they get frustrated with no responses because they will keep on trying challenges until they gain access. Under these circumstances, by allowing ourselves to classify no response as a success, the rate of success for the various styles increases significantly, with style 4 even reaching 90% success. Now imagine the converse scenario, where the CAPTCHA is guarding



| | train | robot | monkey | flower | fish | doll | cat | car | bird | airplane |
|---|---|---|---|---|---|---|---|---|---|---|
| ■ Correct/Total Response | 0.65 | 0.33 | 0.39 | 0.66 | 0.57 | 0.56 | 0.53 | 0.65 | 0.29 | 0.56 |
| ■ No Response/Total Response | 0.26 | 0.41 | 0.43 | 0.27 | 0.31 | 0.29 | 0.25 | 0.34 | 0.45 | 0.34 |

Figure 5.20: Successful Image Identification Across Styles

a resource that has little to no value to a user, for example, a comment section on a popular news article. When a user is faced with a CAPTCHA challenge, they will be disincentivized to post a comment, because now doing so requires more effort than the user is willing to expend. Under these circumstances, a no response can be viewed in a negative light, the same as a failure. If you are a service provider that wants comments, you will be less likely to use a CAPTCHA like this to secure that service. With failure rates above 50%, very few, if any comments will be posted as users will be too frustrated to bother. Ironically enough, if the CAPTCHA is too severe it could possibly make people comment more to complain about the difficulty in solving the CAPTCHA. Laslty, a review of the no response to total response ratio is worthy of review. Style 1 is quickly approaching 50% chance of a response - terrible from a usability perspective. The styles 2-4 all have somewhat reasonable rates between 20%

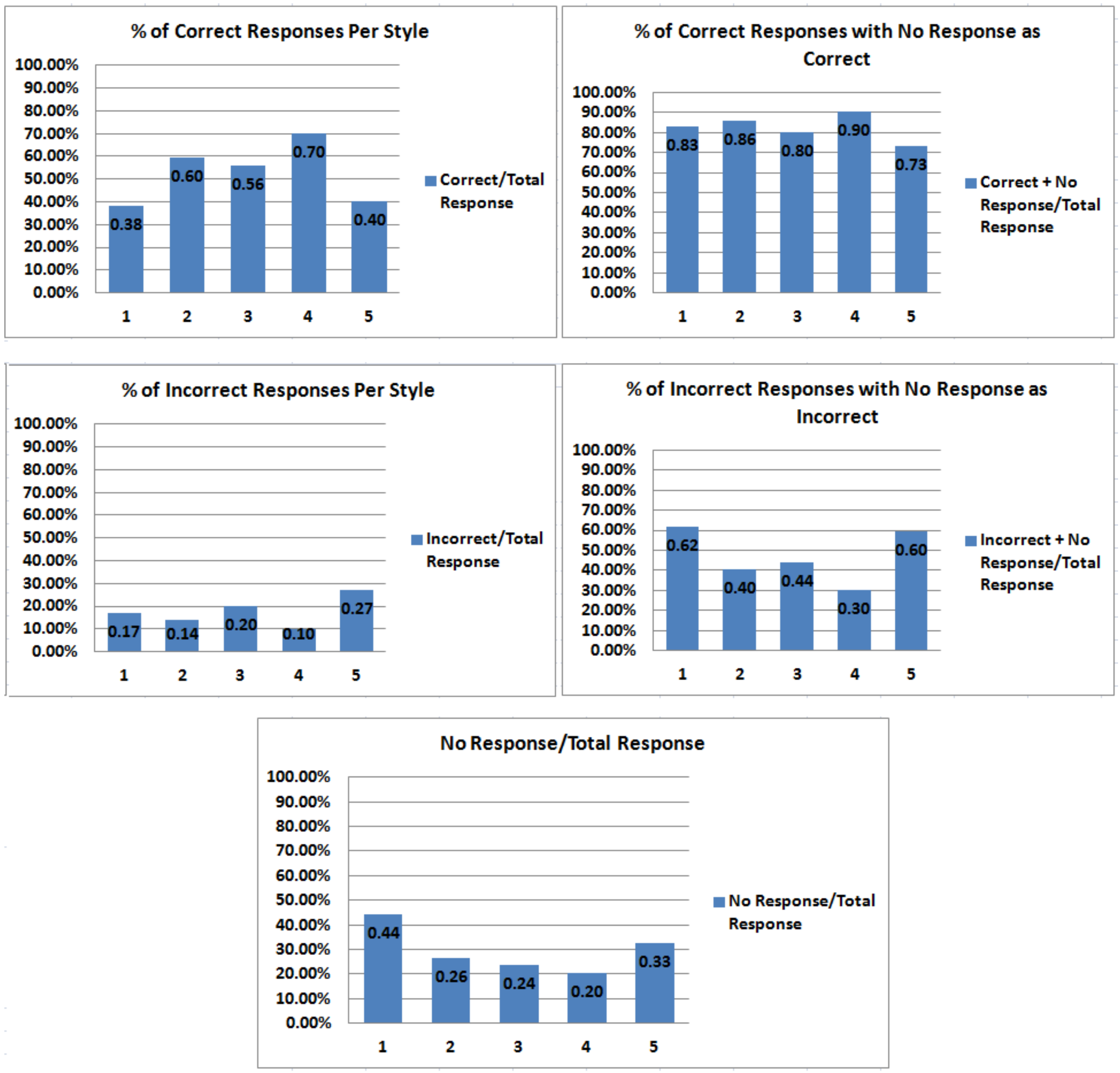


Figure 5.21: Statistical Summary

and 26% respectively, however style 5 has a high rate of no response 33% in spite of the small sample size of tests we recorded for it.

**SIGNAC Study Limitations** All of the statistics generated from the user study presented in the previous section are meant to be somewhat “tongue-in-cheek’, as straight usability tests like the one conducted here do not capture the true nuances of the human response to form security. One can argue that without actually providing a scarce resource, any user behavior that is observed isn’t an accurate representation of how it would be approached in the wild. Without proper context, it is difficult to accurately gauge the performance in usability of any particular CAPTCHA approach. For example, our test users reported that after they answered a number of challenges for each of the different styles, their accuracy improved once they understood what to look for. Another point to consider is when students were given the "pointillism" pointer, that is, to stand further away from the computer monitor to let your eye blend the pixels in the image so its easier to see, reported an increase in the ability to correctly identify categories. CAPTCHA design is equal parts science and art. We have seen from the many examples we offer the tradeoffs within the three major categories, as well as other implementation based security concerns. Every method will have some failures, and the longer a method remains static, the greater the chance a database attack can be generated or a new technique/algorithm developed to defeat it. In this case, it is best to just enumerate all of the possibilities that exist at the time and create a grid that shows the strengths and weaknesses of each method eliminating the ones that fail to meet whatever set of predefined criteria the service provider has decided upon and quantify them accordingly. This is the best advice we can give to an online service provider that wants to generate a form protection method to guard resources from bots and blackhats.

Another point worth noting was that the SIGNAC method used to generate the images was going to have an impact on usability - however this was intentional. We deliberately used a high noise threshold to provide a security guarantee of zero RIS matches, since the images were gathered from online image search engines. This means that the images were significantly more noisy than they needed to be, thus the somewhat unsurprising affected usability performance of the CATPCHA styles. Finally, the users were only able to solve challenges for a short time maybe 5 to 10 minutes. It would be a worthwhile exercise to compare first run results to the same group of users trying the challenges again after they are familiar with all of the styles.

# Chapter 6

# EmojiTCHA - An Emotion based IRC

In this section we will cover the core components and tools used to construct the CAPTCHA challenge that allows for the design to be usable, scalable, and robust - providing a reasonable level of security for online form it is protecting.

## 6.1 Preliminaries & Tools

EmojiTCHA is a complete Image CAPTCHA implementation that evolved out of the work done on SIGNAC. It uses the same image distortion principles as SIGNAC but applies them through the use of destructive and additive filters designed to prevent RIS and CV attacks against an image of face(s) depicting emotion. This is particularly important in the case of EmojiTCHA as it uses automated CV tools to generate the challenges at scale. The following section describes the toolchain used to create the challenges for this CAPTCHA.

**Microsoft Project Oxford** Microsoft's project Oxford is a collection of easy to use artificial intelligence based vision, speech, and language APIs that are cloud accessible and can be used in applications by developers. In our CAPTCHA design, we utilize the Face API and the Emotion API. The Emotion API takes an image as an input, and returns the confidence across a set of emotions for each face in the image, as well as bounding box for the face, using the Face API. If a user has already called the Face API, they can submit the face rectangle as an optional input. The emotions detected are anger, contempt, disgust, fear, happiness, neutral, sadness, and surprise. These emotions are understood to be cross-culturally and universally communicated with particular facial expressions.

Project Oxford's Emotion API is a REST API provided by Microsoft and can be interacted with online. This tool is what provides the critical functionality that delivers the scalablity capabilities within our CAPTCHA design. It provides an automated method to accurately and consistently identify and tag emotions within images that contain people's faces. The output of the Emotion API can then be stored in a database along with the image and subsequently used in a challenge served to a user which asks them to identify the emotions depicted in the image. The power of this service is that the algorithm can easily scale with demand on the CAPTCHA challenge service, e.g. instances can be run in parallel to produce the requested volume of tagged output as required by the challenge service i.e. number of unique challenges that need to be served at a particular rate. In order to prevent the use of this tool against the CAPTCHA, image noise is added to ensure failure to identify emotion or faces on images used in challenges. The process of using and applying image noise will be described in more detail in the GIMP section. More details of the Emotion API can be located at `https://www.projectoxford.ai/emotion`.

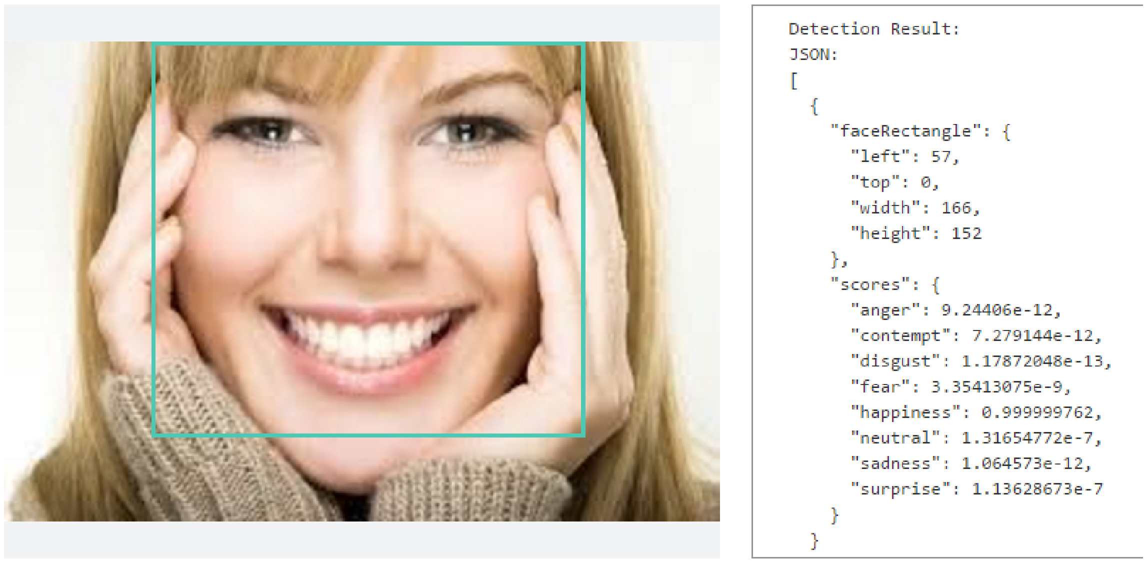
**Emoji Character Set** The emoji character set is a UNICODE character set designed to convey complex ideas and emotions in the form of small ideograms and/or pictograms. The cross-cultural nature of emojis enhances the usability as it removes specific language and alphabets as a barrier to usability. The beauty of using emojis in a CAPTCHA challenge comes from a usability perspective. Our challenge asks a user to match emotions of people in an image with an emoji that conveys the same emotion, providing a solid basis for a simple CAPTCHA challenge task that is easy for humans to understand. Since this character set consists of images instead of text, techniques used to provide noise to the images will also work on the emoji characters, which can be scaled based on font size and noised to thwart attackers further. In our sample design for testing, we have decided to use the twitter emoji set as it has been open sourced for public consumption.

**GIMP: GNU Image Manipulation Program** GIMP is an open source image manipulation program that provides the ability to alter an image based on a set of commands that are applied in a specific sequence. We use this tool to automate the addition of noise and other image alteration/distortion techniques through scripting to change the image used in the challenge. This process is what provides security for the image from attacks by the very tools used to create the CAPTCHA, in essence, providing a "one way" challenge generation function that is very difficult, if not impossible, to reverse.

## 6.2 Methodology

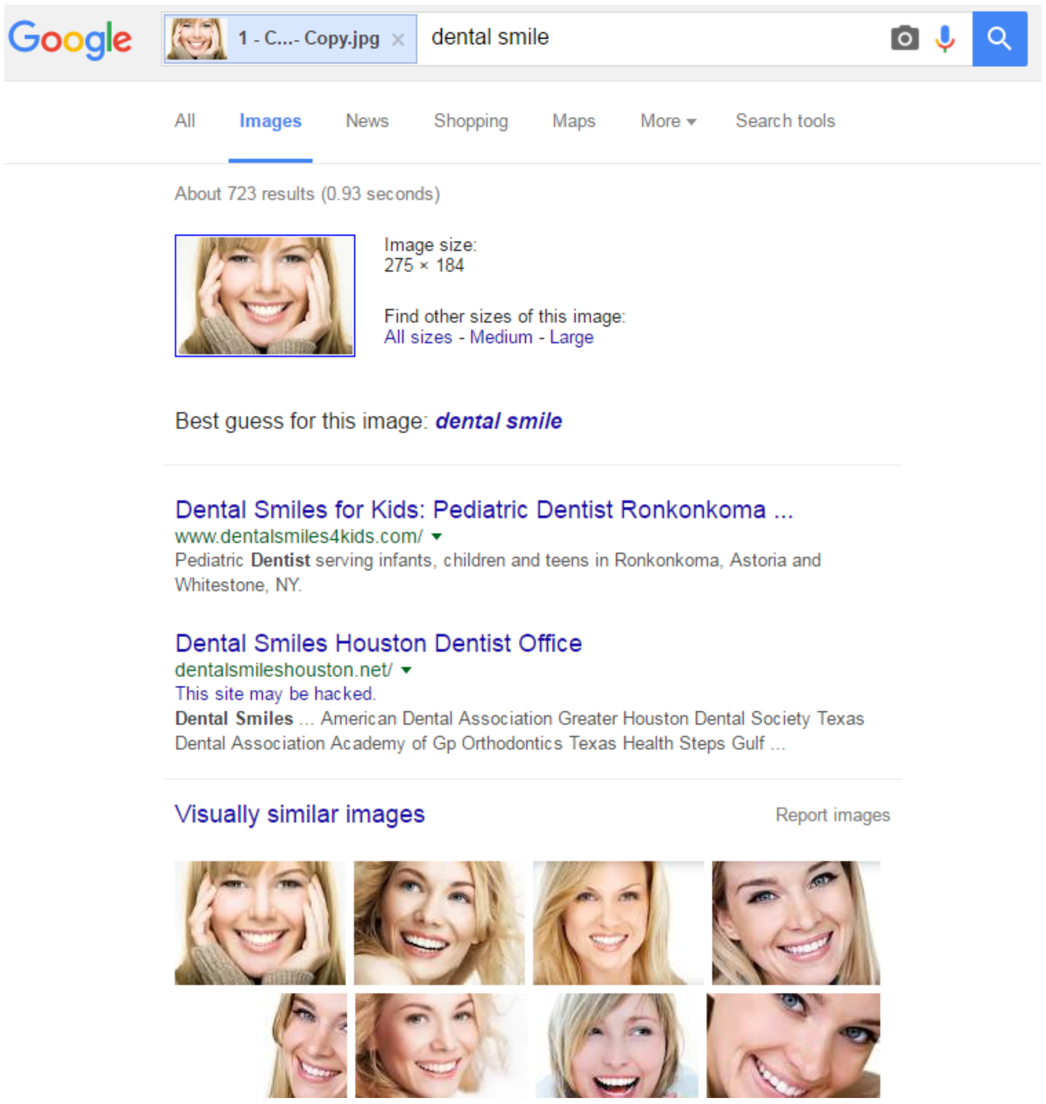
In this section, we will discuss the design choices that were made in order to ensure the usability, scalablity, and robustness of our CAPTCHA while demonstrating the security it provides from potential attacks.

**Image Processing** Figure 6.1 provides an example of the Microsoft Emotion API in action. Using a sample face that is smiling (a depiction of the emotion "happiness"), the API provides the coordinates for a faceRectangle, which is a bounding box based on the area in the image (in pixels) where the Face API detected a human face. It then provides a probability score for each of the eight emotions that it can detect in the form of a percentage out of 100%. These two pieces of information provide the ability to generate a CAPTCHA challenge where the user is asked to identify a face in an image (via the faceRectange), and then answer what emotion that face is expressing. In our implementation, a python script is used to interact with the API online and save the results it returns to a local SQL database, along with the image.



```
Detection Result:
JSON:
[
  {
    "faceRectangle": {
      "left": 57,
      "top": 0,
      "width": 166,
      "height": 152
    },
    "scores": {
      "anger": 9.24406e-12,
      "contempt": 7.279144e-12,
      "disgust": 1.17872048e-13,
      "fear": 3.35413075e-9,
      "happiness": 0.999999762,
      "neutral": 1.31654772e-7,
      "sadness": 1.064573e-12,
      "surprise": 1.13628673e-7
    }
  }
```

Figure 6.1: Example output from Emotion API IMAGE: ©Microsoft Cognitive Services



About 723 results (0.93 seconds)

Image size:
275 × 184

Find other sizes of this image:
All sizes - Medium - Large

Best guess for this image: ***dental smile***

Dental Smiles for Kids: Pediatric Dentist Ronkonkoma ...
www.dentalsmiles4kids.com/
Pediatric **Dentist** serving infants, children and teens in Ronkonkoma, Astoria and Whitestone, NY.

Dental Smiles Houston Dentist Office
dentalsmileshouston.net/
This site may be hacked.
**Dental Smiles** ... American Dental Association Greater Houston Dental Society Texas Dental Association Academy of Gp Orthodontics Texas Health Steps Gulf ...

Visually similar images

Report images

Figure 6.2: Useful information can be produced from default image IMAGE: ©Google Image Search

Figure 6.2 depicts an example of the test image served without noise or filters into Google's reverse image search. Note that the results of the search include the image at other dimensions , a keyword guess for what is depicted in the image (e.g. "dental smile"), and a number of visually similar images that all depict "dental smiles", which if one were to ask a person what emotion was being expressed, most likely "happiness" would be the response. Without introducing noise, distortions and filters to the image, an attacker will be able to answer the challenge question without much difficulty.

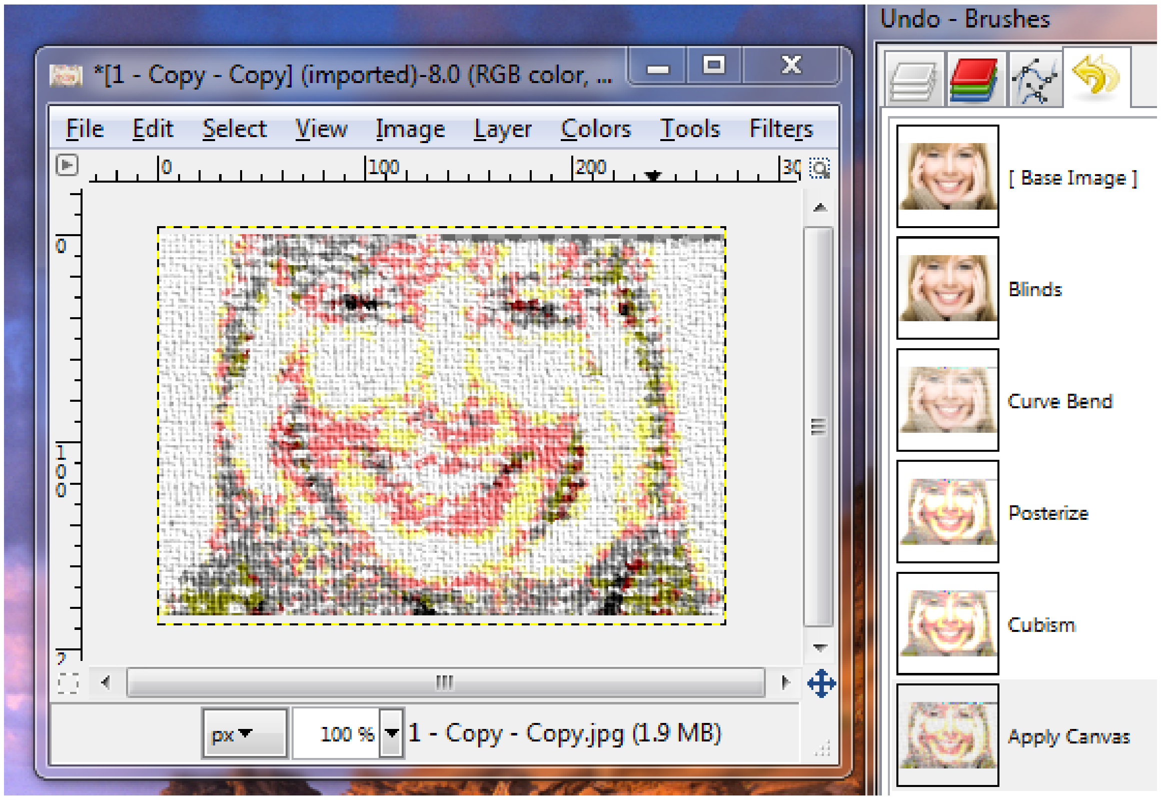


Figure 6.3: Example of series of filters applied to image IMAGE: ©GNU Image Manipulation Program

Figure 6.3 depicts an example chain of filters applied in a specific order to achieve the goal of altering the image enough that RIS, ISS, and CV attacks cannot determine what is depicted in the image. The key is to introduce the minimal amount of distortions such that the tools used to create the challenges are stopped from returning meaningful results. Note that each filter often has multiple parameters that can be adjusted along a range to introduce variability into their output and how they affect the image. For the purposes of our testing, we have determined a series of fixed values for the filters that provide the level of distortion we required to stop the CV attack while still maintaining a reasonable level of usability / ease of understanding the image. Figure 6.4 is an example of an image that meets the RIS, ISS, and CV security requirements.

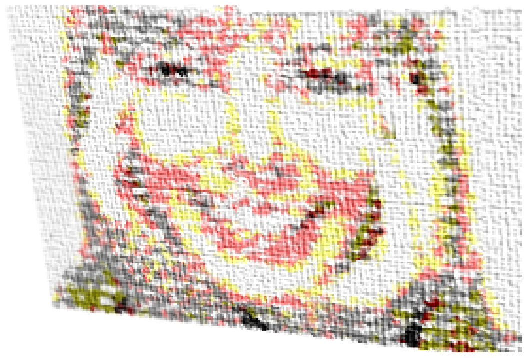


Figure 6.4: Example output image after alterations IMAGE: ©CIMIC Lab - Rutgers University

Figure 6.5 demonstrates how the appropriate level of distorts can “trick” the ISS engine into returning bogus results. Notice that the addition of the “canvas” filter effect influenced the ISS results towards needlepoint/grid based images - none of which focus on facial features. Also notice that there is no keyword provided as well as no images of other sizes. For all intents and purposes, this image is unique to the search engine, despite being indexed and tagged by it.

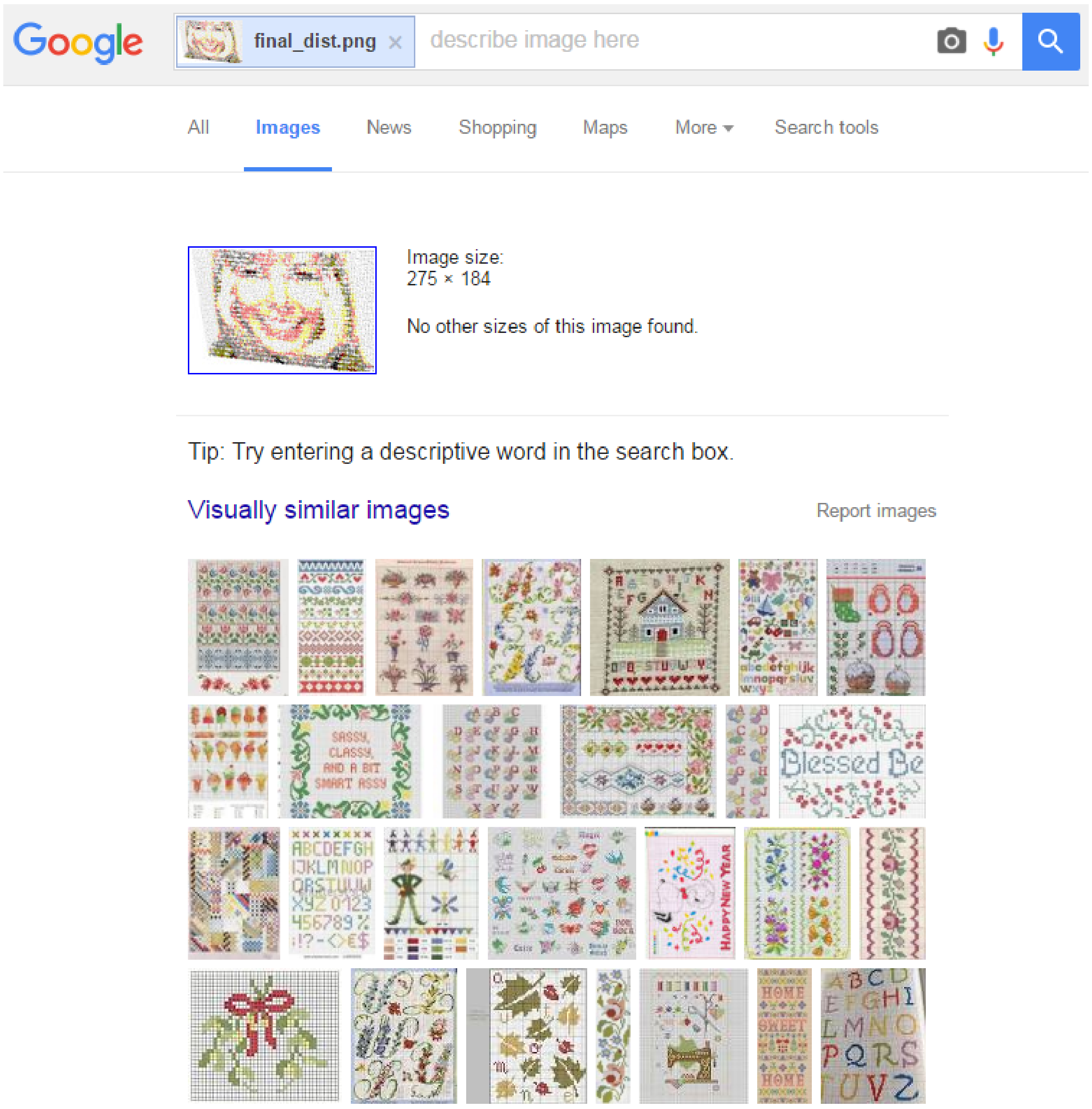


Image size:
275 × 184

No other sizes of this image found.

Tip: Try entering a descriptive word in the search box.

Figure 6.5: No useful information can be extracted from noised image IMAGE: ©Google Image Search

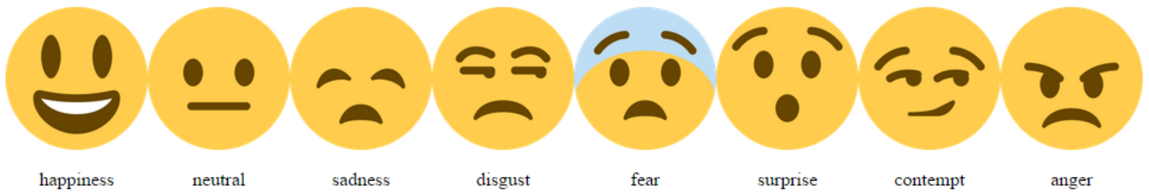


Figure 6.6: Twitter Emojis used to represent the 8 emotions IMAGE: ©Twitter Inc.

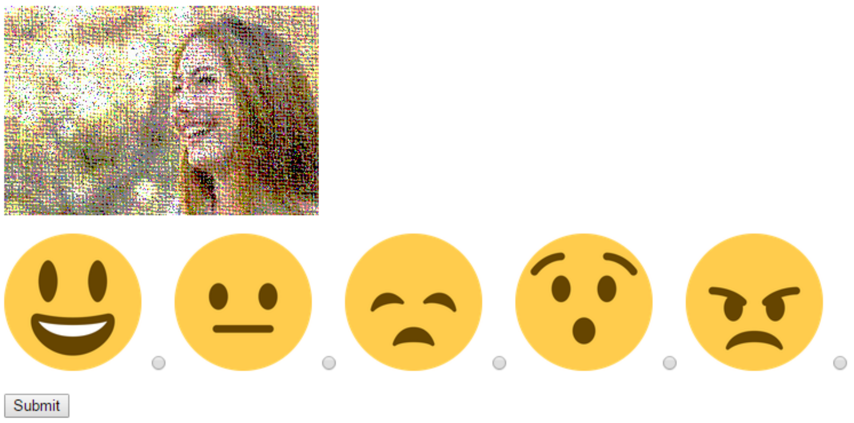


Figure 6.7: Version 2 of EmojiTCHA Challenge IMAGE: ©CIMIC Lab - Rutgers University

Figure 6.6 is an example of the twitter emoji set we used to map to the 8 emotions provided by the Microsoft Emotion API. The initial build of the CAPTCHA included all 8 emotions. Microsoft noted that contempt and disgust were experimental emotions, and thus were usually not read as accurately as the other 6 emotions. After a round of user testing, we decided to decrease the number of emotions that could be selected to the five emotions depicted in figure 6.7 in an effort to remove confusion and increase success. This is an example of the version 2 EmojiTCHA challenge.

## 6.3 CAPTCHA Challenge Generation

This section focuses on the process flow within the toolchain that is used to generate the CAPTCHA challenges. A step by step discussion of the process to generate a unique challenge is included as follows:

1. Gather images involving one or more people whose faces are clearly visible expressing one of the following eight emotions: anger, contempt, disgust, fear, happiness, neutral, sadness, or surprise. These images can be gathered from anywhere - e.g image search engine, downloaded from a camera, etc. They do not need to be tagged as the Emotion API will provide that information.

2. Each image is run through the Microsoft Project Oxford Emotion API to detect the number of faces expressing emotions in the image and the facial expressions that fall into one of the eight emotional categories. If at least one clear face is not found or no emotion can be read from the face by the algorithm, the image is discarded. If at least one clear face expressing one of the eight emotions is found, the image is kept and stored in a database.

3. The output of the Emotion API is recorded in the database along with the stored image. The output from the algorithm includes the face bounding box # (which

face in the image the emotion information is from), the emotion expressed by the face, and the level of confidence as a percentage for the emotion expressed by the face.

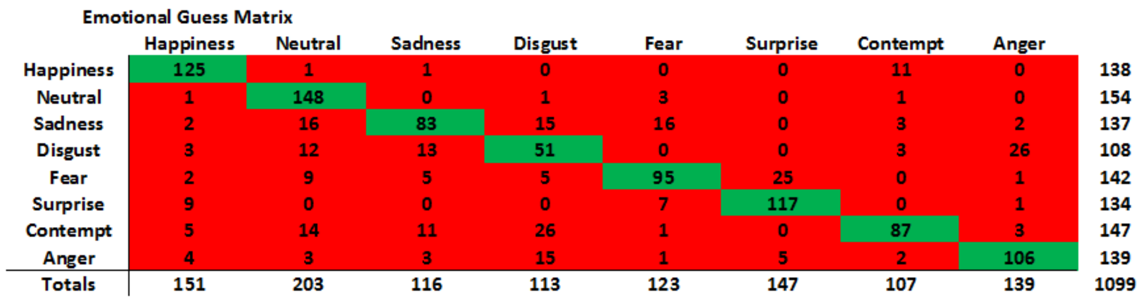
4. The image is then run through a series of filters and manipulations from GIMP to distort/warp/alter the image to protect it against reverse image search attacks and computer vision attacks. The number, type, and values for each of the filters used can be varied at random for each individual image produced to make it very difficult for attackers to filter the alterations. This step is important as it prevents using the tools that generate the challenge from being exploited by attackers. The images can then be tested against image search engines to ensure that no matches are returned. Additional noise is added until the image returns no matches. The final altered image is then stored in the database with the corresponding information used to create it.

5. A set of emojis are selected where a subset of these emojis match the emotion recorded from the Emotion API, with the remaining emojis not matching the emotions i.e. they would be incorrect/nonexistent responses.

6. The challenge is generated and the user is presented with an image of one or more distorted faces and a corresponding set of emojis. The user must match the the correct emoji to the correct facial expression in the image to complete the challenge. The CAPTCHA evaluates the correctness of the response by querying the database for facial bounding box information along with its corresponding expressed emotion.

## 6.4 EmojiTCHA Usability Study

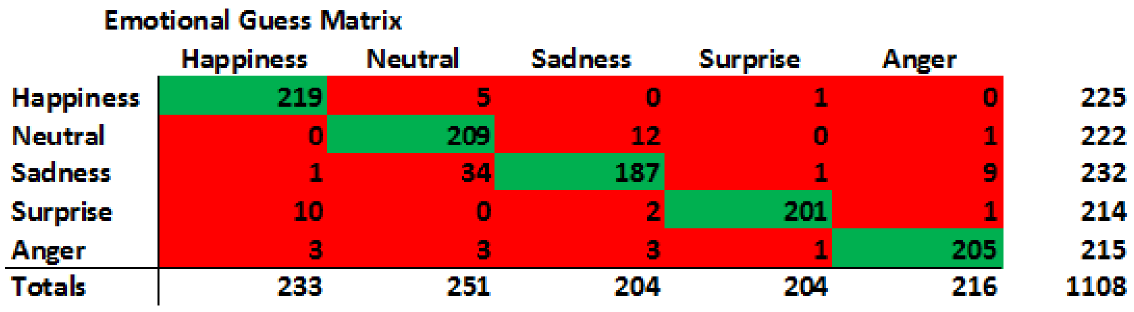
The goal of this section is to demonstrate the effectiveness and ease of use of the new CAPTCHA EmojiTCHA. We conducted user trials with 30 participants and asked them to solve as many challenges as they could in 10 minutes. The first run included all eight emotions from the emotion engine. The user was served a challenge at random and asked to match the corresponding emoji to the emotion depicted on the face displayed. Each of the emotion categories had 10 images that were selected by hand and are able to be tagged and identified by the Emotion API. The image filters were applied at random until the image no longer returned a match from the Emotion API, thus some images were more distorted than others.



Emotional Guess Matrix

| | Happiness | Neutral | Sadness | Disgust | Fear | Surprise | Contempt | Anger | |
|---|---|---|---|---|---|---|---|---|---|
| Happiness | 125 | 1 | 1 | 0 | 0 | 0 | 11 | 0 | 138 |
| Neutral | 1 | 148 | 0 | 1 | 3 | 0 | 1 | 0 | 154 |
| Sadness | 2 | 16 | 83 | 15 | 16 | 0 | 3 | 2 | 137 |
| Disgust | 3 | 12 | 13 | 51 | 0 | 0 | 3 | 26 | 108 |
| Fear | 2 | 9 | 5 | 5 | 95 | 25 | 0 | 1 | 142 |
| Surprise | 9 | 0 | 0 | 0 | 7 | 117 | 0 | 1 | 134 |
| Contempt | 5 | 14 | 11 | 26 | 1 | 0 | 87 | 3 | 147 |
| Anger | 4 | 3 | 3 | 15 | 1 | 5 | 2 | 106 | 139 |
| Totals | 151 | 203 | 116 | 113 | 123 | 147 | 107 | 139 | 1099 |

Figure 6.8: Emotion Matrix for Run 1



Emotional Guess Matrix

| | Happiness | Neutral | Sadness | Surprise | Anger | |
|---|---|---|---|---|---|---|
| Happiness | 219 | 5 | 0 | 1 | 0 | 225 |
| Neutral | 0 | 209 | 12 | 0 | 1 | 222 |
| Sadness | 1 | 34 | 187 | 1 | 9 | 232 |
| Surprise | 10 | 0 | 2 | 201 | 1 | 214 |
| Anger | 3 | 3 | 3 | 1 | 205 | 215 |
| Totals | 233 | 251 | 204 | 204 | 216 | 1108 |

Figure 6.9: Emotion Matrix for Run 2

Figure 6.8 demonstrates the emotional guess matrix for the first run with the complete set of emotions. The totals on the horizontal axis represent the number of times a challenge with the correct response being the emotion in green was served whereas the totals on the vertical axis represent the number of times that a particular emotion was given as a response for a challenge with the correct answer in green. Figure 6.9 demonstrates the same ideas but for the second run with the reduced set of emotions as options.

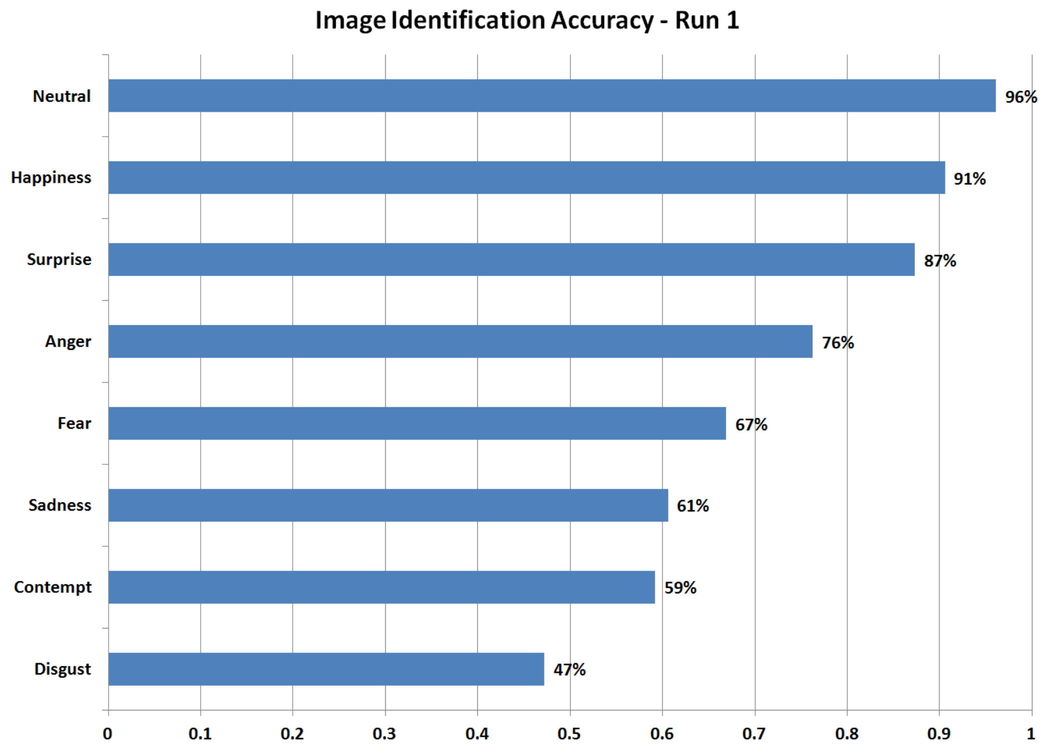


Figure 6.10: Image Identification - Run 1

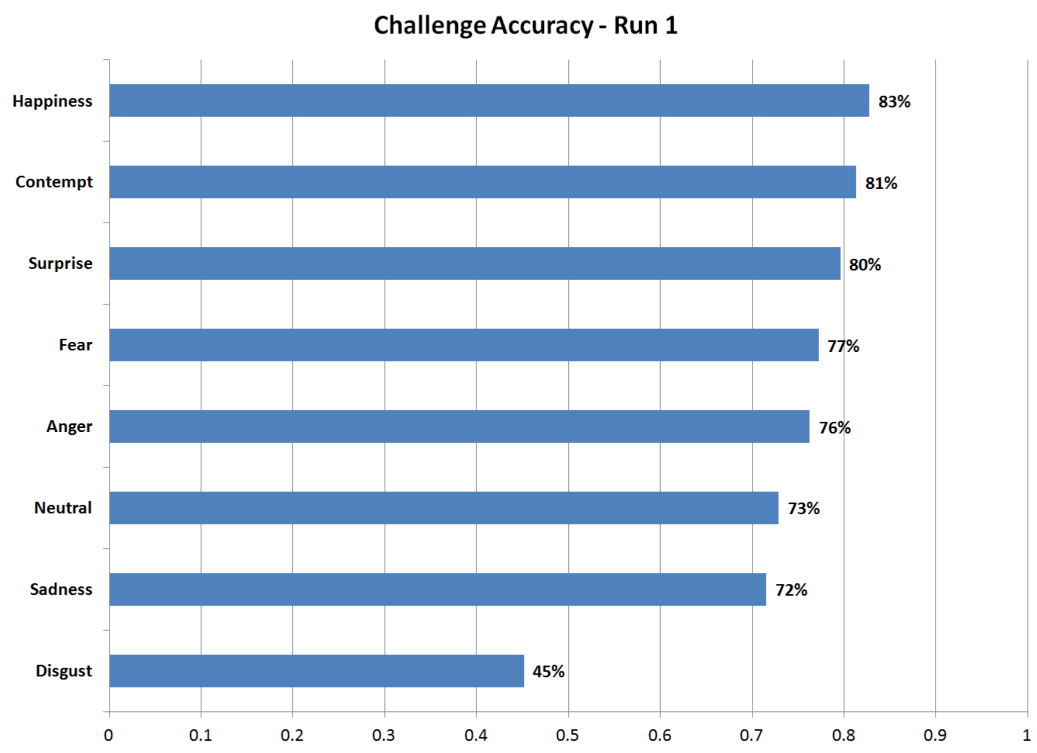


Figure 6.11: Challenge Response Accuracy - Run 1

Figures 6.12 and 6.13 demonstrate the percentages that were obtained from an analysis of the matrix for each of the categories. The emotions from best performing to worse performing on image identification are: neutral(96%), happiness(91%), surprise(87%), anger(76%), fear(67%), sadness(61%), contempt(59%), and disgust(47%). The emotions from best to worse performing on challenge response are: happiness(83%), contempt(81%), surprise(80%), fear(77%), anger(76%), neutral(73%), sadness(72%), and disgust(45%). Note that disgust was the worst performing emotion in both cases. It was expected that the more abstract of the universal emotions might be more difficult to discern for humans, eg. disgust and contempt. For image identification, these emotions performed the worst, scoring significantly lower than top three emotions. Strangely enough, for challenge accuracy contempt was the second best emotion recognized, although the scores for the challenges were somewhat lower than the scores for image identification, they were much more consistent across emotions, with disgust being the outlier. One aspect is the overlap of emotions that are consistently mistaken for another emotion that may appear similar. For example, we see that disgust was mistaken for anger 26 times. It is easy to imagine that a disgusted face can take a similar shape to an angry one. We also see this in contempt and anger being mistaken for disgust as well at 26 and 15 times respectively. More user testing will need to be conducted so that any set of emotions served to the user in a challenge will be ones that are not easily mistaken for each other. However, this can also provide a way to make it more difficult for machines - if a competing emotion detection algorithm is ranking a facial expression it is possible that it will score and categorize it differently than the MS Emotion API. Further work is needed to determine the best approach.

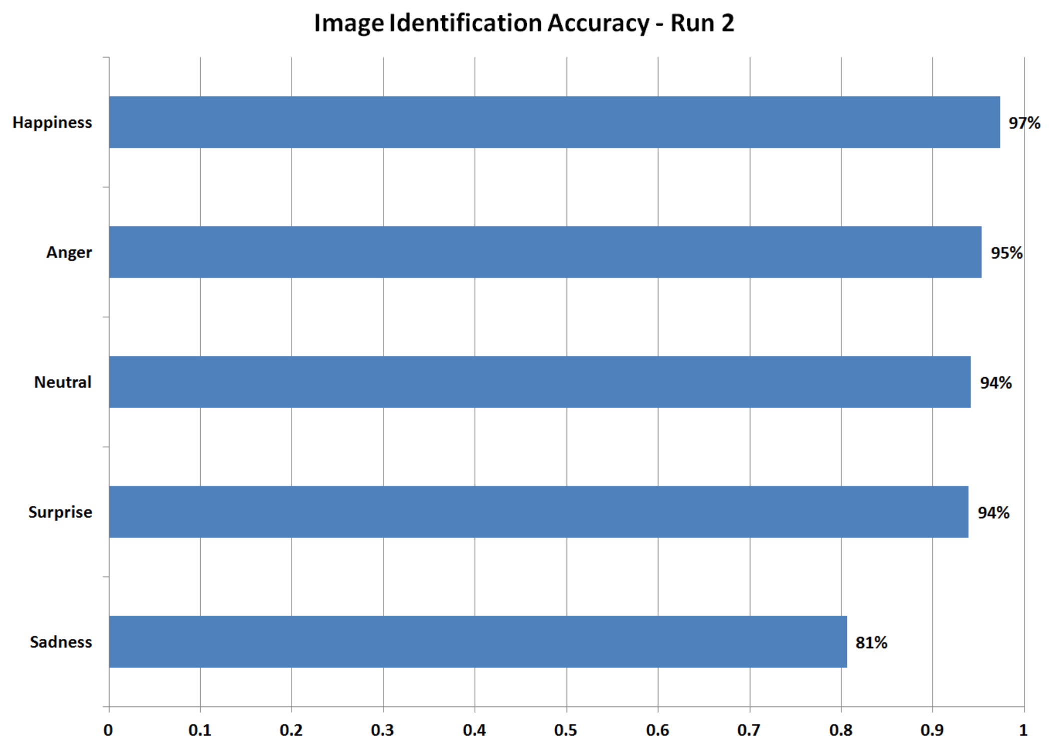


Figure 6.12: Image Identification - Run 2

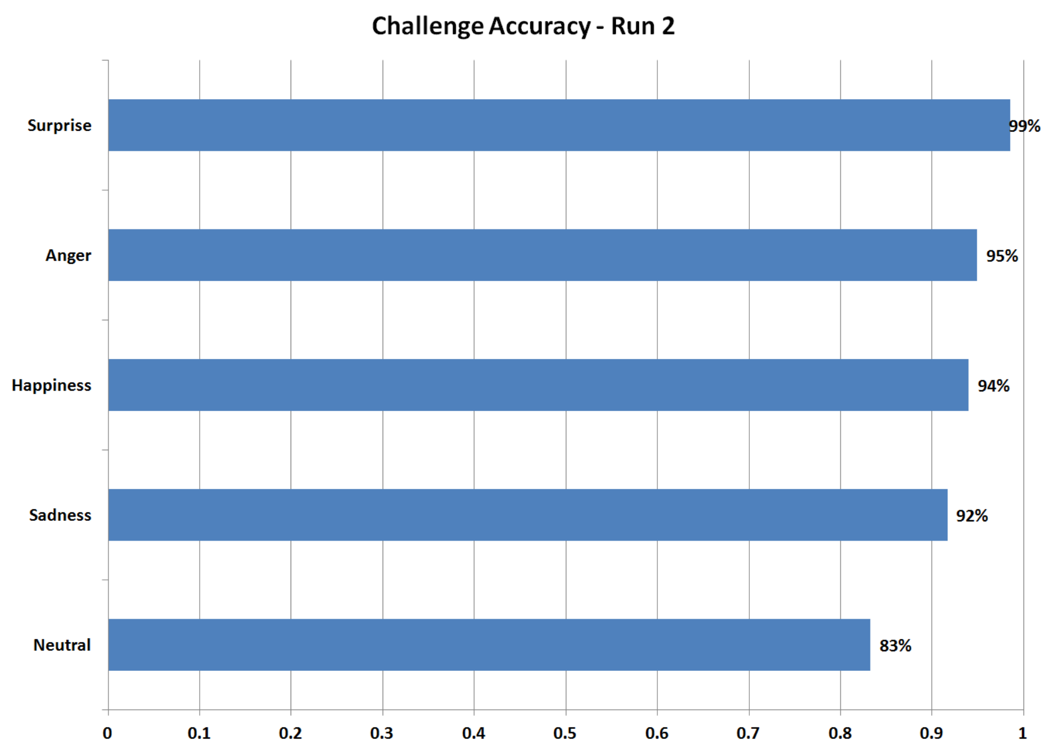


Figure 6.13: Challenge Response Accuracy - Run 2

Figures 6.12 and 6.13 demonstrate the percentages that were obtained from an analysis of the matrix for each of the categories. The emotions from best performing to worse performing on image identification are: happiness(97%), anger(95%), neutral(94%), surprise(94%), sadness(81%). The emotions from best to worse performing on challenge response are: surprise(99%), anger(95%), happiness(94%), sadness(92%), neutral(83%). The performance for all scores in both categories increased significatly in limiting the number of choices for the user to select.

### 6.4.1 EmojiTCHA Study Limitations

The images that were chosen for use in the challenges were hand curated to ensure that the desired emotion was demonstrated in the image. Work is currently in progress to tune the image scraper and the Emotion API checker to accept an "emotion" threshold score as a percentage to ensure that there is a high degree of confidence in any particular emotion expressed in a face. Any images that have emotions that register below the threshold can be discarded accordingly. Additional work must be done to determine the optimal "co-emotions" to display with one another to ensure there is minimal mixup by between the emotions displayed on the screen. Our goal is to reintroduce all 8 emotions but have them operate more effectively by optimizing what other emotions they are displayed with. These will all be addressed in future iterations of the study.

## 6.5 Limitations

In our current implementation, we only use a single emotion from a single face per challenge. Mutli-face, multi emotion challenges are currently under development. We are also currently developing ordinality rules for filter application to minimize the number of rounds of filter applications that are required to ensure the security guarantees that CV attacks and RIS attacks will not be successful. The test images used in the experiments for the user study only provide the security guarantee for emotion API attacks - they do not ensure RIS or ISS attacks are not successful, although many of the images do indeed stop these attacks in their current form.

## 6.6 Future Work

In the future, additional work around new form types and challenge questions will be experimented with. For example, testing out multiple emotions in a single image and having a user identify all of the emotions - a multi-answer CAPTCHA. Another example would be asking the user to identify the opposite emotion of that depicted in an image (e.g. pick sad if the image is showing a happy face). Additional work around creating a challenge with an "emotional mix" where a random set of 4 or 5 choices are selected for the challenge from the 8 possible choices. Finally, experimenting with a “not here” answer may be worthwhile to increase the security against a random guess being correct.

# Chapter 7

# Future Work & Conclusions

This section speculates on the future work that can be conducted as a result of this work and the conclusions drawn from the body of work conducted for this dissertation.

## 7.1 Summary of Contributions

In this dissertation, a number of contributions to the field of CAPTCHA research have been made. A broad review of the ideas and concepts surrounding CAPTCHA design and illustrations of how selecting and implementing a hard AI problem can present a number of challenges for the CAPTCHA author to overcome if they want their design to score highly within the three major design criteria. We present two attack methods that are effective against IRC's that do not use noise/distortions/filters to protect their images from computer vision and other types of attacks. We also demonstrate a security method named SIGNAC which utilized noise to allow image CAPTCHAs to use images from tagged online databases and remain safe from reverse image search and other tools that allow attackers to circumvent performing the challenge task. Finally, we also present a new image CAPTCHA, called EmojiTCHA, that is a fully automated, scalable, and robust image CAPTCHA that relies on contextual data within the image as its central challenge question. Thanks to new tools made available to the public, we have been able to obtain the “holy grail” of building an IRC. User studies have also been conducted on our defensive methods to ensure that usability remains a top priority while also simultaneously providing a strong guarantee of security under a variety of attack circumstances.

## 7.2 Future Work

Since CAPTCHA attack and design is an eternal arms race, there are plenty of avenues worthy of exploration in the future as the tools and algorithms available continue to evolve and improve.

### 7.2.1 Online Tools for Attacks

One area worth exploring in the future is that of online image annotation services. Although the idea is not new, these recently developed tools have evolved out of the significant leaps forward that CV and machine learning techniques have made that are now being applied to datasets at scale. The ability to leverage the web to gather large numbers of images makes developing these types of systems easier, as there is more data to use in training and testing advanced models and methods of image analysis. Another point worth considering is the rise in the availability of high quality image sensors propagating throughout the population via smartphone adoption. When combined with the rise in the use of social media, which provides constant streams of new data that contain important metadata produced by humans (e.g. image tags, hashtags, scene descriptions etc.) these datastreams/datasets provide excellent opportunities to develop online computer vision and machine learning interfaces that could potentially be used to defeat image recognition CAPTCHAs. For example, IBM Alchemy leverages the neural networks of Watson to perform advanced image scene analysis and can provide probabilistic guesses as to what is depicted in an image. There are number of these services beginning to emerge online, and many have strikingly accurate results and have been demonstrated to be effectively leveraged to attack IRC's. As with any online tool, they are still susceptible to noised images, as they were not designed to handle this type of challenge and assume a clean input to operate upon. However, a deep learning model with a large training dataset and an accurate idea of how noise and/or what noise algorithms are used on a particular image may be able to eventually discern the image as well as contextual clues and be able to solve IRC challenges. This is one avenue we plan on pursuing in the future for attacks against IRCs that use noised images.

### 7.2.2 Cryptocurrency as a CAPTCHA

As CAPTCHA increasingly becomes defined in economic terms and as CV algorithms improve, it may be worthwhile to investigate the use of a cryptocurrency as a way to "paywall" scarce services that are to be made available online. A user who wishes to access the account/information/whatever is being protected can use their CPU/GPU/etc. to perform some task that is useful to the administrator of the scarce resource. If there is no task, the generation of a monetary unit that can be spent elsewhere will also suffice for the purposes of the exchange of goods and services. In order to prevent market flooding and ambushing of services, the difficulty of mining a particular CAPTCHA coin could be increased at a rate that would allow the administrator to throttle the available currency in the marketplace to match the rate of service he can provide. The blockchain data structure which is implemented in the form of a distributed database, provides many unique opportunities to prevent abuse of online services through demonstrating that work or a particular transaction that serves as "work" within the ecosystem has taken place and that this work cannot be altered or revised. This is a potentially exciting avenue of research as computer hardware has advanced to the point where large scale public online ecosystems could utilize computational resources to solve interesting problems in a distributed fashion as a method of verification of work and blockchains could be used to verify the work. One aspect to this "quantification" of work worth considering is that it essentially removes the reverse Turing test aspect of CAPTCHA and simply treats the service in terms of units of work - it no longer matters if it is done by a human or a bot. This can be viewed from the economic perspective as a direct result of solving farms - and also ironically enough, provides a solution to the problem of sentient bots - as now the online service is governed by cost functions and alternative methods to dissuade abuse of the service must be found.

### 7.2.3 Biometric CAPTCHAs

As discerning between human and bot becomes increasingly difficult, turning to new sensors and emerging methods to track human bodies may provide the best and easiest path forward in CAPTCHA design. Fingerprint readers, cameras, and fitness trackers have become somewhat ubiquitous in the modern age as people begin to use trackers to measure and quantify various aspects of their lives. There is potential for these to be used in various methods of CAPTCHA to ensure that a human is present and not a bot just performing a clever attack on a sensor to act human. This avenue is worth exploring in the future as smartphones and intelligent wearables (e.g. fitness trackers, smartwatches, Google Glass etc.) begin to become part of everyday life. However, using these methods presents a number of security and privacy challenges that will need to be addressed accordingly. One challenging issue with biometrics is the security implications surrounding a breach of the database used for biometric sensor verification. Since biometric data is extremely personal and unique (at least for now you cannot change your fingerprints, irises, vein paths etc.) losing this data means that it can no longer be easily used to verify you, as you can only lose it once to have it lost forever. Part of the reason CAPTCHA development is important is that it provides a somewhat anonymous way of verifying that a user is human without delving too deep into unique aspects of that humanity.

# Bibliography

[1] R. Aadhirai, P. Kumar, and S. Vishnupriya. Image captcha: Based on human understanding of real world distances. In *Intelligent Human Computer Interaction (IHCI), 2012 4th International Conference on*, pages 1–6, Dec 2012.

[2] A. E. Ahmad, J. Yan, and W.-Y. Ng. Captcha design: Color, usability, and security. *IEEE Internet Computing*, 16(2):44–51, 2012.

[3] L. V. Ahn, M. Blum, N. J. Hopper, and J. Langford. Captcha: using hard ai problems for security. In *Proceedings of the 22nd international conference on Theory and applications of cryptographic techniques*, EUROCRYPT'03, pages 294–311, Berlin, Heidelberg, 2003. Springer-Verlag.

[4] A. Almazyad, Y. Ahmad, and S. Kouchay. Multi-modal captcha: A user verification scheme. In *Information Science and Applications (ICISA), 2011 International Conference on*, pages 1–7, April 2011.

[5] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(4):509 –522, apr 2002.

[6] J. Blocki, M. Blum, and A. Datta. Gotcha password hackers! In *Proceedings of the 2013 ACM Workshop on Artificial Intelligence and Security*, AISec '13, pages 25–34, New York, NY, USA, 2013. ACM.

[7] E. Bursztein, J. Aigrain, A. Moscicki, and J. C. Mitchell. The end is nigh: Generic solving of text-based captchas. In *8th USENIX Workshop on Offensive Technologies (WOOT 14)*, San Diego, CA, Aug. 2014. USENIX Association.

[8] E. Bursztein, S. Bethard, C. Fabry, J. C. Mitchell, and D. Jurafsky. How good are humans at solving captchas? a large scale evaluation. In *Proceedings of the 2010 IEEE Symposium on Security and Privacy*, SP '10, pages 399–413, Washington, DC, USA, 2010. IEEE Computer Society.

[9] E. Bursztein, M. Martin, and J. Mitchell. Text-based captcha strengths and weaknesses. In *Proceedings of the 18th ACM Conference on Computer and Communications Security*, CCS '11, pages 125–138, New York, NY, USA, 2011. ACM.

[10] G. Chechik, V. Sharma, U. Shalit, and S. Bengio. Large scale online learning of image similarity through ranking. *J. Mach. Learn. Res.*, 11:1109–1135, Mar. 2010.

[11] K. Chellapilla, K. Larson, P. Simard, and M. Czerwinski. Designing human friendly human interaction proofs (hips). In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '05, pages 711–720, New York, NY, USA, 2005. ACM.

[12] E. Y. Chen, L.-S. Huang, O. J. Mengshoel, and J. D. Lohn. Darwin: A ground truth agnostic captcha generator using evolutionary algorithm. In *Proceedings of the Companion Publication of the 2014 Annual Conference on Genetic and Evolutionary Computation*, GECCO Comp '14, pages 165–166, New York, NY, USA, 2014. ACM.

[13] M. Chew and J. Tygar. Image recognition captchas. In K. Zhang and Y. Zheng, editors, *Information Security*, volume 3225 of *Lecture Notes in Computer Science*, pages 268–279. Springer Berlin Heidelberg, 2004.

[14] G. Cybenko. Deep learning of behaviors for security. In *Proceedings of the 2015 ACM International Workshop on International Workshop on Security and Privacy Analytics*, IWSPA '15, pages 1–1, New York, NY, USA, 2015. ACM.

[15] W. Daher and R. Canetti. Posh: A generalized captcha with security applications. In *Proceedings of the 1st ACM Workshop on Workshop on AISec*, AISec '08, pages 1–10, New York, NY, USA, 2008. ACM.

[16] R. Datta, D. Joshi, J. Li, and J. Z. Wang. Image retrieval: Ideas, influences, and trends of the new age. *ACM Comput. Surv.*, 40(2):5:1–5:60, May 2008.

[17] R. Datta, J. Li, and J. Wang. Exploiting the human-machine gap in image recognition for designing captchas. *Information Forensics and Security, IEEE Transactions on*, 4(3):504 –518, sept. 2009.

[18] R. Datta, J. Li, and J. Z. Wang. Imagination: a robust image-based captcha generation system. In *Proceedings of the 13th annual ACM international conference on Multimedia*, MULTIMEDIA '05, pages 331–334, New York, NY, USA, 2005. ACM.

[19] M. Egele, L. Bilge, E. Kirda, and C. Kruegel. Captcha smuggling: Hijacking web browsing sessions to create captcha farms. In *Proceedings of the 2010 ACM Symposium on Applied Computing*, SAC '10, pages 1865–1870, New York, NY, USA, 2010. ACM.

[20] A. S. El Ahmad, J. Yan, and L. Marshall. The robustness of a new captcha. In *Proceedings of the Third European Workshop on System Security*, EUROSEC '10, pages 36–41, New York, NY, USA, 2010. ACM.

[21] J. Elson, J. Douceur, J. Howell, and J. Saul. Asirra: a captcha that exploits interest-aligned manual image categorization. In *Proceedings of the 14th ACM conference on Computer and communications security*, CCS '07, pages 366–374, New York, NY, USA, 2007. ACM.

[22] P. Faymonville, K. Wang, J. Miller, and S. Belongie. Captcha-based image labeling on the soylent grid. In *Proceedings of the ACM SIGKDD Workshop on Human Computation*, HCOMP '09, pages 46–49, New York, NY, USA, 2009. ACM.

[23] C. A. Fidas, A. G. Voyiatzis, and N. M. Avouris. On the necessity of user-friendly captcha. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pages 2623–2626, New York, NY, USA, 2011. ACM.

[24] C. Fritsch, M. Netter, A. Reisser, and G. Pernul. Attacking image recognition captchas: a naive but effective approach. In *Proceedings of the 7th international conference on Trust, privacy and security in digital business*, TrustBus'10, pages 13–25, Berlin, Heidelberg, 2010. Springer-Verlag.

[25] H. Gao, D. Yao, H. Liu, X. Liu, and L. Wang. A novel image based captcha using jigsaw puzzle. In *Computational Science and Engineering (CSE), 2010 IEEE 13th International Conference on*, pages 351–356, Dec 2010.

[26] T. Gao and D. Koller. Discriminative learning of relaxed hierarchy for large-scale visual recognition. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2072–2079, 2011.

[27] N. Gelernter and A. Herzberg. Tell me about yourself: The malicious captcha attack. In *Proceedings of the 25th International Conference on World Wide Web*, WWW '16, pages 999–1008, Republic and Canton of Geneva, Switzerland, 2016. International World Wide Web Conferences Steering Committee.

[28] P. Giura, I. Murynets, R. Piqueras Jover, and Y. Vahlis. Is it really you?: User identification via adaptive behavior fingerprinting. In *Proceedings of the 4th ACM Conference on Data and Application Security and Privacy*, CODASPY '14, pages 333–344, New York, NY, USA, 2014. ACM.

[29] E. Gladkikh, K. Nikolaev, and M. Nikitin. Localized captcha testing on users and farms. In *Proceedings of the 23rd International Conference on World Wide Web*, WWW '14 Companion, pages 277–278, New York, NY, USA, 2014. ACM.

[30] P. Golle. Machine learning attacks against the asirra captcha. In *Proceedings of the 15th ACM Conference on Computer and Communications Security*, CCS '08, pages 535–542, New York, NY, USA, 2008. ACM.

[31] I. J. Goodfellow, Y. Bulatov, J. Ibarz, S. Arnoud, and V. D. Shet. Multi-digit number recognition from street view imagery using deep convolutional neural networks. *CoRR*, abs/1312.6082, 2013.

[32] R. Gossweiler, M. Kamvar, and S. Baluja. What's up captcha?: A captcha based on image orientation. In *Proceedings of the 18th International Conference on World Wide Web*, WWW '09, pages 841–850, New York, NY, USA, 2009. ACM.

[33] G. Gu. Machine learning meets social networking security: Detecting and analyzing malicious social networks for fun and profit. In *Proceedings of the 5th ACM Workshop on Security and Artificial Intelligence*, AISec '12, pages 1–2, New York, NY, USA, 2012. ACM.

[34] C. J. Hernandez-Castro, A. Ribagorda, and Y. Saez. Side-channel attack on the humanauth captcha. In *Security and Cryptography (SECRYPT), Proceedings of the 2010 International Conference on*, pages 1–7, July 2010.

[35] F. Inc. Funcaptcha: most secure captcha that's easy for humans @ONLINE. `http://www.funcaptcha.com/`.

[36] M. Inc. Documentation center - imnoise. http://www.mathworks.com/help/images/ref/imnoise.html, sep 2014.

[37] M. Inc. How-old.net@ONLINE. `http://how-old.net/`, December 2015.

[38] N. Inc. Nucaptcha – adaptive captcha authentication @ONLINE. `http://www.nucaptcha.com/`.

[39] N. Inc. *Getting Started with NuPIC*, 2008. This is an electronic document. Date of publication: [September 2008]. Version 1.2.1.

[40] N. Inc. *Numenta Vision Toolkit Tutorial*, 2009. This is an electronic document. Date of publication: [August 6, 2009].

[41] M. Jakobsson. Captcha-free throttling. In *Proceedings of the 2Nd ACM Workshop on Security and Artificial Intelligence*, AISec '09, pages 15–22, New York, NY, USA, 2009. ACM.

[42] M.-F. Jian, H.-K. Chu, R.-R. Lee, C.-L. Ku, Y.-S. Wang, and C.-Y. Yao. Emerging images synthesis from photographs. In *ACM SIGGRAPH 2013 Posters*, SIGGRAPH '13, pages 97:1–97:1, New York, NY, USA, 2013. ACM.

[43] Y. Jing and S. Baluja. Pagerank for product image search. In *Proceedings of the 17th international conference on World Wide Web*, WWW '08, pages 307–316, New York, NY, USA, 2008. ACM.

[44] Y. Jing, H. Rowley, J. Wang, D. Tsai, C. Rosenberg, and M. Covell. Google image swirl: a large-scale content-based image visualization system. In *Proceedings of the 21st international conference companion on World Wide Web*, WWW '12 Companion, pages 539–540, New York, NY, USA, 2012. ACM.

[45] C. I. Lab. Computer vision online demos. https://www.cs.cmu.edu/ cil/v-demos.html, jun 2005.

[46] S. R. Lang and N. Williams. Impeding captcha breakers with visual decryption. In *Proceedings of the Eighth Australasian Conference on Information Security - Volume 105*, AISC '10, pages 39–46, Darlinghurst, Australia, Australia, 2010. Australian Computer Society, Inc.

[47] J. Li and J. Z. Wang. Real-time computerized annotation of pictures. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(6):985–1002, June 2008.

[48] S. Li, S. A. H. Shah, M. A. U. Khan, S. A. Khayam, A.-R. Sadeghi, and R. Schmitz. Breaking e-banking captchas. In *Proceedings of the 26th Annual Computer Security Applications Conference*, ACSAC '10, pages 171–180, New York, NY, USA, 2010. ACM.

[49] A. Linn. Happy? sad? angry? this microsoft tool recognizes emotions in pictures@ONLINE. `http://blogs.microsoft.com/next/2015/11/11/happy-sad-angry-this-microsoft-tool-recognizes-emotions-in-pictures/`, November 2015.

[50] C. T. Lopez. Army's mind lab able to decode brain waves@ONLINE. `http://www.army.mil/article/158256`, November 2015.

[51] D. G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the International Conference on Computer Vision-Volume 2 - Volume 2*, ICCV '99, pages 1150–, Washington, DC, USA, 1999. IEEE Computer Society.

[52] N. A. S. M. Tariq Banday. Image flip captcha. *ISeCure, The ISC International Journal of Information Security*, 1(2):105–123, 2009.

[53] J. Markoff. Computer eyesight gets a lot more accurate @ONLINE. `http://bits.blogs.nytimes.com/2014/08/18/computer-eyesight-gets-a-lot-more-accurate/?_r=0`, August 2014.

[54] P. Matthews, A. Mantel, and C. C. Zou. Scene tagging: image-based captcha using image composition and object relationships. In *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security*, ASIACCS '10, pages 345–350, New York, NY, USA, 2010. ACM.

[55] M. Mehrnejad, A. Bafghi, A. Harati, and E. Toreini. Multiple seimcha: Multiple semantic image captcha. In *Internet Technology and Secured Transactions (ICITST), 2011 International Conference for*, pages 196–201, Dec 2011.

[56] B. Mehta, S. Nangia, M. Gupta, and W. Nejdl. Detecting image spam using visual features and near duplicate detection. In *Proceedings of the 17th International Conference on World Wide Web*, WWW '08, pages 497–506, New York, NY, USA, 2008. ACM.

[57] N. J. Mitra, H.-K. Chu, T.-Y. Lee, L. Wolf, H. Yeshurun, and D. Cohen-Or. Emerging images. In *ACM SIGGRAPH Asia 2009 Papers*, SIGGRAPH Asia '09, pages 163:1–163:8, New York, NY, USA, 2009. ACM.

[58] M. Mohamed, N. Sachdeva, M. Georgescu, S. Gao, N. Saxena, C. Zhang, P. Kumaraguru, P. C. van Oorschot, and W.-B. Chen. A three-way investigation of a game-captcha: Automated attacks, relay attacks and usability. In *Proceedings of the 9th ACM Symposium on Information, Computer and Communications Security*, ASIA CCS '14, pages 195–206, New York, NY, USA, 2014. ACM.

[59] J.-M. Morel and G. Yu. Asift: A new framework for fully affine invariant image comparison. *SIAM J. Img. Sci.*, 2(2):438–469, Apr. 2009.

[60] G. Mori and J. Malik. Recognizing objects in adversarial clutter: breaking a visual captcha. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 1, pages I–134 – I–141 vol.1, june 2003.

[61] M. Motoyama, K. Levchenko, C. Kanich, D. McCoy, G. M. Voelker, and S. Savage. Re: Captchas: Understanding captcha-solving services in an economic context. In *Proceedings of the 19th USENIX Conference on Security*, USENIX Security'10, pages 28–28, Berkeley, CA, USA, 2010. USENIX Association.

[62] J. Pita, R. John, R. Maheswaran, M. Tambe, R. Yang, and S. Kraus. A robust approach to addressing human adversaries in security games. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems - Volume 3*, AAMAS '12, pages 1297–1298, Richland, SC, 2012. International Foundation for Autonomous Agents and Multiagent Systems.

[63] S. A. Ross, J. A. Halderman, and A. Finkelstein. Sketcha: a captcha based on line drawings of 3d models. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 821–830, New York, NY, USA, 2010. ACM.

[64] Y. Shen, R. Ji, D. Cao, and M. Wang. Hacking chinese touclick captcha by multi-scale corner structure model with fast pattern matching. In *Proceedings of the 22Nd ACM International Conference on Multimedia*, MM '14, pages 853–856, New York, NY, USA, 2014. ACM.

[65] V. Shet. Street view and recaptcha technology just got smarter @ONLINE. `http://googleonlinesecurity.blogspot.com.es/2014/04/street-view-and-recaptcha-technology.html`, April 2014.

[66] S. Shirali-Shahreza and M. Shirali-Shahreza. Categorizing captcha. In *Proceedings of the 4th ACM workshop on Security and artificial intelligence*, AISec '11, pages 107–108, New York, NY, USA, 2011. ACM.

[67] A. Shrivastava, T. Malisiewicz, A. Gupta, and A. A. Efros. Data-driven visual similarity for cross-domain image matching. In *Proceedings of the 2011 SIGGRAPH Asia Conference*, SA '11, pages 154:1–154:10, New York, NY, USA, 2011. ACM.

[68] V. Srikanth, C. Vishwanathan, U. Asati, and N. C. S. N. Iyengar. Think-an image based captcha mechanism (testifying human based on intelligence and knowledge). In *Proceedings of the International Conference on Advances in Computing, Communication and Control*, ICAC3 '09, pages 421–424, New York, NY, USA, 2009. ACM.

[69] D. Tsai, Y. Jing, Y. Liu, H. Rowley, S. Ioffe, and J. Rehg. Large-scale image annotation using visual synset. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 611–618, 2011.

[70] R. ur Rahman, D. Tomar, and S. Das. Dynamic image based captcha. In *Communication Systems and Network Technologies (CSNT), 2012 International Conference on*, pages 90–94, May 2012.

[71] S. Vikram, Y. Fan, and G. Gu. Semage: a new image-based two-factor captcha. In *Proceedings of the 27th Annual Computer Security Applications Conference*, ACSAC '11, pages 237–246, New York, NY, USA, 2011. ACM.

[72] O. Warner. Kittenauth captcha@ONLINE. `http://thepcspy.com/kittenauth/`, December 2015.

[73] S. S. Woo, B. Kim, W. Jun, and J. Kim. 3doc: 3d object captcha. In *Proceedings of the 23rd International Conference on World Wide Web*, WWW '14 Companion, pages 397–398, New York, NY, USA, 2014. ACM.

[74] J. Yan and A. S. E. Ahmad. Captcha security: A case study. *IEEE Security & Privacy*, 7(4):22–28, 2009.

[75] J. Yan and A. S. El Ahmad. A low-cost attack on a microsoft captcha. In *Proceedings of the 15th ACM Conference on Computer and Communications Security*, CCS '08, pages 543–554, New York, NY, USA, 2008. ACM.

[76] J. Yan and A. S. El Ahmad. Captcha robustness: A security engineering perspective. *Computer*, 44:54–60, February 2011.

[77] B. Zhu, J. Yan, G. Bao, M. Yang, and N. Xu. Captcha as graphical passwords a new security primitive based on hard ai problems. *Information Forensics and Security, IEEE Transactions on*, 9(6):891–904, June 2014.

[78] B. B. Zhu, J. Yan, Q. Li, C. Yang, J. Liu, N. Xu, M. Yi, and K. Cai. Attacks and design of image recognition captchas. In *Proceedings of the 17th ACM Conference on Computer and Communications Security*, CCS '10, pages 187–200, New York, NY, USA, 2010. ACM.