No Bot Expects the DeepCAPTCHA!
Introducing Immutable Adversarial Examples with Applications to CAPTCHA

Margarita Osadchy
Department of Computer Science, University of Haifa, Mount Carmel, Haifa 31905, Israel
rita@cs.haifa.ac.il

Julio Hernandez-Castro
School of Computing, University of Kent, Canterbury CT2 7NF, Kent, UK
J.C.Hernandez-Castro@kent.ac.uk

Stuart Gibson
School of Physical Sciences, University of Kent, Canterbury CT2 7NF, Kent, UK
s.j.gibson@kent.ac.uk

Orr Dunkelman
Department of Computer Science, University of Haifa, Mount Carmel, Haifa 31905, Israel
orrd@cs.haifa.ac.il

Daniel Pérez-Cabo
Gradiant, Campus Universitario de Vigo, 36310 Pontevedra, Spain
dpcabo@gradiant.org

Abstract

Recent advances in Deep Learning (DL) allow for solving complex AI problems that used to be very hard. While this progress has advanced many fields, it is considered to be bad news for CAPTCHAs (Completely Automated Public Turing tests to tell Computers and Humans Apart), the security of which is based on the hardness of learning problems.

In this paper we introduce DeepCAPTCHA, a new and secure CAPTCHA scheme based on *adversarial examples*, an inherit limitation of the current Deep Learning networks.[1] These adversarial examples are constructed inputs, computed by adding a small and specific perturbation called *adversarial noise* to correctly classified items, causing the targeted DL network to misclassify them. We show that plain adversarial noise is insufficient to achieve secure CAPTCHA schemes, which leads us to introduce *immutable adversarial noise* — an adversarial noise resistant to removal attempts.

We implement a proof of concept system and its analysis shows that the scheme offers high security and good usability compared to the best existing CAPTCHAs.

1 Introduction

CAPTCHAs are traditionally defined as automatically constructed problems, very difficult to solve for artificial intelligence (AI) algorithms, but easy for humans. Due to progress in AI, an increasing number of CAPTCHA designs have become ineffective, as the underlying AI problems became solvable by algorithmic tools. Specifically, recent advances in Deep Learning (DL) reduced the gap between human and machine ability in solving problems that have been typically used in CAPTCHAs. This breakthrough in AI led some researches to believe that DL would lead to the "end" of CAPTCHAs [4, 14].

Despite having achieved human-competitive accuracy in complex tasks such as speech processing and image recognition, DL still has some important shortcomings with regards to human ability. Our proposal exploits these shortcomings to reliably distinguish between hu-

[1] We just became aware of a recent paper [28] speculating about the possibility of using adversarial examples for CAPTCHAs. We have developed the idea independently throughout 2015, building the theory, implementing a proof of concept system, and running experiments. Our work was submitted to USENIX Security in February 2016, before [28] was made public.

mans and bots in a way that is secure, user-friendly, and scalable.

Our scheme is based on an interesting and unexpected limitation of Deep Learning, discovered recently in [30], called adversarial examples. These are constructed inputs computed by adding a small and specific perturbation, called *adversarial noise*, to correctly classified items. The aim is to cause the network to misclassify these perturbed examples, with high confidence. Although discovered in the context of Deep Learning, this phenomenon was observed also in other classifiers and it was shown to be related to an inherit property of low-capacity classifiers (which are classifiers with low flexibility, e.g., linear classifiers) to be overconfident when extrapolating away from the separation boundary [11]. Existing DL networks are piecewise linear, which is the most likely reason for their vulnerability to adversarial examples, according to [15]. We note (and later discuss) that high-capacity models (such as RBF) are more robust to adversarial examples, but they are unable to cope with large-scale tasks, for example those involving more than 1000 categories.

We propose to use adversarial examples for CAPTCHA generation within an object classification framework, involving a large number of classes. Adversarial examples are appealing for CAPTCHA applications as they are very difficult for DL and easy for humans (adversarial noise tends to be small). However, to be useful in CAPTCHA settings, the adversarial noise should have an additional property that has not been addressed yet. The noise should be resistant to filtering or any other attacks that attempt to remove it.

We introduce the notion of *immutable adversarial noise*, an adversarial noise that cannot be removed by a preprocessing algorithm. We analyze existing methods for adversarial noise generation and demonstrate that they do not posses this important property. We then develop a new method for creating *immutable adversarial noise*. We also analyze various attacks that attempt to cancel this noise, and show that their success rate is very low.

In particularly, in the context of CAPTCHAs, one needs to maintain usability, i.e., the adversarial noise should be small, to keep the image recognizable by humans. This makes the problem of designing an immutable adversarial noise even more challenging, as small magnitude noise (due to this usability requirement) is harder to protect against removal attacks (security requirement). Note that for other cyber security applications, not involving human evaluation of the input, immutability could be achieved more easily by increasing the magnitude of the adversarial noise.

1.1 Our Contribution

This paper proposes DeepCAPTCHA – a new concept of CAPTCHA generation that employs specifically designed adversarial noise to deceive Deep Learning classification tools. The noise is kept small to not significantly affect the recognition ability of humans, but is made resistant to removal attacks.

Previous methods for adversarial noise generation lack the robustness to filtering or any other attacks that attempt to remove the adversarial noise. We are the first to address this problem and we solve it by generating an *immutable* adversarial noise with emphasis on image filtering. We analyze the security of our construction against a number of complementary attacks and show that it is highly robust to all of them.

Finally, we introduce the first proof-of-concept implementation of DeepCAPTCHA. Our results show that the approach has merit in terms of both security and usability.

2 Related Work

We start our discussion with reviewing the most prominent work in CAPTCHA generation and then we turn to the Deep Learning area, focusing on methods for creating adversarial examples.

2.1 A brief introduction to CAPTCHAs

Since their introduction as a method of distinguishing humans from machines [33], CAPTCHAs (also called inverse Turing tests [22]) have been widely used in Internet security for various tasks. Their chief uses are mitigating the impact of Distributed Denial of Service (DDoS) attacks, slowing down automatic registration of free email addresses or spam posting to forums, and also as a defense against automatic scraping of web contents [33].

Despite their utility, current CAPTCHA schemes are often loathed by humans as they present an additional obstacle to accessing internet services and many schemes suffer from very poor usability [5, 37].

2.1.1 Text Based Schemes

The first generation of CAPTCHAs used deformations of written text. This approach has now became less popular due to its susceptibility to segmentation attacks [36]. In response, some developers increased distortion levels, using methods such as character overlapping, which increases security [7]. Unfortunately, such measures have also resulted in schemes that are frequently unreadable by humans. We note that some text-based implementations are susceptible to general purpose tools [4].

2.1.2 Image Based Schemes

Motivated by the vulnerability of text based schemes, image based CAPTCHAs have been developed, following the belief that these were more resilient to automated attacks [9, 10, 13, 39]. For example, early text based versions of the reCAPTCHA [34] system were superseded by a combined text and image based approach. However, the new scheme was also subsequently attacked in [14]. The most recent version of the system is NoCAPTCHA, shown with its predecessors in Figure 1.

An alternative approach is CORTCHA (Context-based Object Recognition to Tell Computers and Humans Apart) that claims resilience to machine learning attacks [39]. This system uses the contextual relationships between objects in an image, in which users are required to re-position objects to form meaningful groupings. This task requires a higher level reasoning in addition to simple object recognition.

2.1.3 Alternative Schemes

Considerable effort is currently being invested in novel ways of implementing secure and usable CAPTCHAs. Two of the most popular research themes are video-based CAPTCHAs such as NuCAPTCHA [24], and game-based CAPTCHAs [21]. The former have generally shown inadequate security levels so far [3, 35]. The latter designs are in general inspired by the AreYouHuman CAPTCHA [1]. One of the most interesting proposals in this group is [21], an example of a DGC (Dynamic Cognitive Game) CAPTCHA that has the additional advantage of offering some resistance to relay attacks, and a high usability. Unfortunately, in its current form, it is vulnerable to automated dictionary attacks. One can also argue that recent developments in game playing by computers, that match or improve human abilities by using deep reinforcement learning [20], question the prospects of future game based proposals. Finally, a number of puzzle-based CAPTCHAs that seemingly offered some promise have recently been subjected to devastating attacks [16].

2.1.4 Deep Learning Attacks

The general consensus within the cyber security community is that CAPTCHAs that simultaneously combine good usability and security are becoming increasingly hard to design, due to potential threats from bots armed with Deep Learning [4, 14] capabilities. This has led to the popularity of Google's NoCAPTCHA re-CAPTCHA despite its violation of a number of important CAPTCHA and general security principles.[2]

[2]For example, the **P** in CA**P**TCHA stands for Public, and NoCAPTCHA inner functioning is not public, based on the time-dishonored concept of "security by obscurity" by employing heavily obfuscated Javascript code.

2.2 Deep Learning and Adversarial Examples

Deep Learning networks are designed to learn multiple levels of representation and abstraction for different types of data such as images, video, text, and speech. Convolutional Neural Networks (CNNs) are DL algorithms that have been successfully applied to image classification tasks since 1989 [18].

The development of AlexNet [17] introduced a number of improvements to the classical architecture of CNN's (notably "dropout" and the ReLu activation function [8]) that have been used to train the CNN on GPU's. This seminal work triggered the return of CNNs to the forefront of machine learning research, and their extensive use in current leading work within the field. Many modifications of the basic concept have been proposed in recent years, allowing classification improvements for increasingly larger sets (e.g., [6, 12, 27, 29, 31, 38])

Hereafter, we will use the terms CNN (convolutional neural network) and DL (deep learning) network interchangeably.

2.2.1 Adversarial Examples – Foundations

Adversarial examples were introduced in [30] as inputs, constructed by adding a small tailored noise component to correctly classified items that cause the DL network to misclassify them, with high confidence. The existence of such unexpected phenomenon was explained in [30] by shortcomings in the generalization abilities of DL algorithms.

A more theoretical explanation of this instability phenomenon was suggested in [11]. Namely, they showed a fundamental limit on the robustness of low-capacity classifiers (e.g., linear, quadratic) to adversarial perturbations. They suggested expressing this limitation in terms of a distinguishability measure between classes, which depends on the chosen family of classifiers. Specifically, for linear classifiers the distinguishability is defined as the distance between the means of the two classes. For quadratic classifiers, it is defined as the distance between the second order moments of the classes [11]. Other classification models or multi-class settings have not been addressed yet, but it was noted that higher capacity models with highly non-linear decision boundaries are significantly more robust to adversarial inputs.

Neural Networks can learn different capacity models, ranging from linear to highly non-linear. DL architectures are considered to have very large capacity, allowing highly non-linear functions to be learned. However, training such DL networks is hard and doing it efficiently

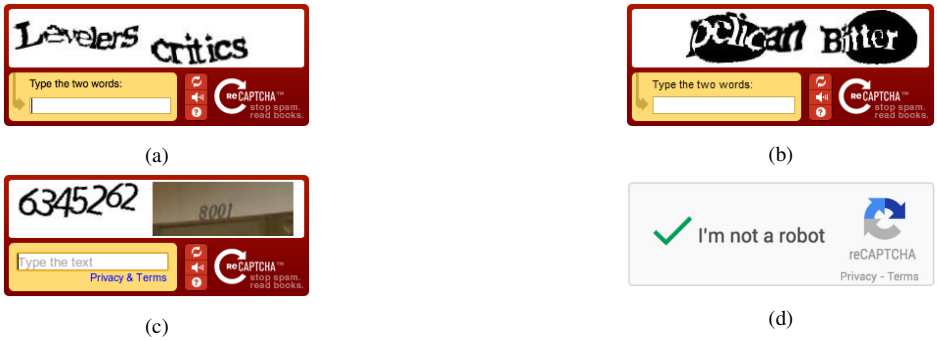


Figure 1: Example of reCAPTCHA evolution through time, from the initial designs (a) to increasingly more complex and robust ones (b), including some incorporating images (c), and ending with the current (d) NoCAPTCHA version.

remains an open problem. The only architectures (and activation functions) that are currently practical to train over complex problems have a piecewise linear nature which is the most likely reason for their vulnerability to adversarial examples [15].

It is important to note that the recently derived upper bound on the robustness to adversarial perturbations [11] is valid for **all** classifiers of the class (e.g., all linear classifiers), independently of the training procedure. A similar conclusion was derived empirically in [15], for different variants of DL networks and verified also in our experiments (see Section 5).

Previous work [11, 15] outlined a number of solutions for adversarial instability. One of them was to switch to highly non-linear models, for instance, RBF Neural Networks or RBF Support Vector Machines. These are shown to be significantly more robust to adversarial samples but are extremely hard to train and slow to run and are currently considered impractical for large-scale problems. Another proposition was to train DL networks on adversarial examples. This made the network robust against the examples in the training set and improved the overall generalization abilities of the network [11, 15]. However, it did not resolve the problem as other adversarial samples could be efficiently found for the new network, especially in large-scale problems.

2.2.2 Adversarial Examples – Constructions

On the practical side, different types and techniques for constructing adversarial inputs have been proposed in recent works [15, 23, 25, 30]. The approach in [23] causes a neural network to classify an input as a legitimate object with high confidence, even though it is perceived as random noise or a simple geometric pattern by a human. The method in [25] focuses on making the adversarial noise affect only a small portion of the image. Such noise has little effect on the recognition by humans, but is adversarial to a DL network. Unfortunately, both of these methods are not suitable for CAPTCHA design. The former will obviously suffer from poor usability, while the latter will offer very poor security. Localized noise, particularly in images of white digits on a black background (as exemplified in [25]) can be easily removed by a number of simple image processing algorithms, notably spatial filters.

The techniques proposed in [15, 30] compute an image-dependent and small-magnitude noise component such that, when added to the original image, results in a perturbation that is not perceptible to the human eye but causes the DL network to completely misclassify the image with high confidence. Consequently, these methods have a great potential to be useful in CAPTCHA generation.

2.2.3 Adversarial Examples – Immutability

The question of robustness of adversarial noise to the attacks that remove the adversarial effect has not been addressed yet. The main reason for this, is that the phenomenon was discovered in machine learning community with the goal of demonstrating that DL tools do not fully understand the classification task at hand, but rather follow an instruction based approach [15].

In order to use adversarial noise in CAPTCHAs or other security applications, it should be immutable to removal attacks which can employ alternative tools and, in particular, image processing methods. We show here that none of the existing methods for adversarial example construction are sufficiently robust to such attacks. Even though the approach in [25] proposed a construction of adversarial examples in a computer security context, it also lacks this necessary property.

2.2.4 Adversarial Examples in CAPTCHA

The current state of technology does not offer a solution for a large scale (+1000 categories) multi-class recog-

nition problem that is robust to adversarial examples. There is also some evidence supporting the additional desirable property (for our purposes) that adversarial examples generalize well, and are consistently difficult to classify across different network architectures and even for dissimilar initialization values and sizes [30]. Moreover, for some types of classifiers, the adversarial instability is an inherit limitation. These limitations, combined with the fact that adversarial noise could be made almost imperceptible to the human eye, render the idea of using adversarial examples as the basis for new CAPTCHA challenges very appealing.

3 Immutable Adversarial Noise Generation

Adversarial noise is designed to deceive DL networks. However, one can preprocess the network inputs in a attempt to remove the adversarial perturbation. Hence, in a computer security setting, adversarial noise must withstand any preprocessing that cancels the adversarial effect. We show in the following that previous methods for generating adversarial examples do not provide a sufficient security in this respect.

We introduce the concept of *Immutable Adversarial Noise* (IAN), as an adversarial perturbation that withstands cancellation attempts. Since the nature of the cancellation attempts depends on the security target, we instantiate the concept of IAN for CAPTCHAs. In particular, we explicitly define the requirements for creating IAN for CAPTCHA generation, then we analyze the previous algorithms for adversarial examples and show that they do not meet these requirements. Finally, we present our new algorithm for generation of IANs that satisfy the new requirements.

3.1 Requirements for IAN in CAPTCHA

An algorithm for an immutable adversarial noise construction for CAPTCHA needs to meet the following requirements:

1. **Adversarial:** The added noise should be successful in deceiving the targeted system in 100% of cases.[3]

2. **Robust:** The added noise should be very difficult to remove by any computationally efficient means; for example by filtering or by ML approaches.

3. **Perceptually Small:** The added noise should be small enough to not interfere with a successful recognition of the image contents by humans.

4. **Efficient:** The algorithm should be computationally efficient, to allow for the generation of possibly millions of challenges per second, with adequate but moderate hardware resources.

A basic requirement for CAPTCHAs is that challenges do not repeat and are not predictable (i.e., guessing one out of m possible answers should succeed with probability $1/m$). Hence, the source of , used for generating adversarial examples, should be **bottomless** and uniform. An algorithm that can create an adversarial example out of an arbitrary image, together with such a bottomless and uniform source can generate a bottomless and uniform set of challenges.

3.2 Test Bed Details

In all our experiments throughout the paper, we based the creation of adversarial examples on the CNN-F deep network from [6], implemented in MatConvNet [32]. We used the ILSVRC-2012 database [26], containing 1000 categories of natural images. The DL network was trained on the training set of the ILSVRC-2012 database, and the adversarial examples were created using its validation set.

Using non-overlapping sets for CNN training and adversarial example generation is dictated by security reasons. The training set of a network is assumed to be known by the attacker or can otherwise be deduced from the network parameters (such methods are not within the scope of this paper) and used to learn the adversarial noise for the training images and each other class. If this was the case, the attacker would have the mapping from adversarial images to the true label.

All experiments were conducted on a Linux 3.13.0 Ubuntu machine with an Intel(R) QuadCore(TM) i3-4160 CPU @ 3.60GHz, 32GB RAM, with two GTX 970 GPU cards, using MATLAB 8.3.0.532 (R2014a).

3.3 Previous Methodologies for Generating Adversarial Examples

We briefly introduce in the following the previously known methods for adversarial noise generation, and discuss why they do not meet the above requirements.

Our idea is to use images that are easily recognized by humans but are adversarial to DL algorithms. Consequently, methods that cause a DL network to classify images of noise or geometric patterns as objects [23] are not adequate for our goal. We also rule out the method from [25] due to its obvious lack of robustness to simple local image filtering.

[3] The 100% requirement could be slightly relaxed in large systems.

The Optimization Method Szegedy et al. [30] introduced the concept of adversarial examples in the context of DL networks and proposed a straightforward way of computing them using the following optimization problem:

$$\arg\min_{\Delta_I} \|\Delta_I\|^2 \ \ s.t.\ Net(I+\Delta_I) = C_d \qquad (1)$$

where I is the original input from class C_i, Δ_I is the adversarial noise, *Net* is the DL classification algorithm, and C_d is the deceiving class, such that $C_d \neq C_i$. Once the adversarial noise is found, the corresponding adversarial image is constructed as $adv(I, C_d, p) = I + \Delta_I$, where p is the target confidence level of the *Net* in misclassifying $adv(I, C_d, p)$ as coming from C_d.

We implemented and tested this optimization method as shown in Eq. (1), over a set of 1000 images. Fast computation of adversarial examples is an essential requirement for any viable CAPTCHA deployment, since it will need to generate millions of challenges per second. The optimization method described here is too slow, hence for practical purposes we limited the number of iterations to a fixed threshold (it stops when this limit is reached). This, however, resulted in a failure to produce the desired class or confidence level (of 80%) in some cases. The time statistics of the experiment and the success rate are shown in Table 1.

Based on these results, we can conclude that the optimization algorithm is not suitable for our needs: it is computationally expensive and it does not converge in some cases. The inefficiency of this generation method has been reported before, and is explicitly mentioned in [15, 30].

The computation of adversarial examples needs to be fast and immutable. In this context, *immutable* refers to the difficulty of inverting the noise addition to the original image. We tested the immutability of the adversarial examples created by the optimization method to a simple filtering attack. We tried various filters (and parameters), and found that the median filter of size 5x5 was the most successful in reverting the noise. After applying it to 1000 adversarial images created by the optimization algorithm, it was possible to correctly classify 16.2% of the examples, as reported in Table 1. Thus, the immutability of this method to a removal attack is low and it fails by quite some margin to provide the required security level.[4]

The Fast Gradient Sign Method A much faster method for generating adversarial examples was proposed in [15]. The approach is called the *fast gradient sign method* and it computes the adversarial noise as follows:

$$\Delta_I = \varepsilon \cdot sign(\nabla_I J(W, I, C_d))$$

where $\nabla_I J(W, I, C_d)$ is the gradient of the cost function J (used to train the network) with respect to the input I, W are the trained network parameters, and ε is a constant which controls the amount of noise inserted.

The fast gradient sign method computes the gradient of the target class with respect to the input image, and adds its sign multiplied by some constant to the original image. The bigger this constant is, the larger the adversarial effect and the degradation of the image are. In our implementation, we observed that the best approach to get an adversarial image with high confidence rates while keeping the noise minimal is by running a noise generation step with a small ε, in several iterations. The price of this approach is a small increase in the running time.

We tested the fast gradient sign method, and verified that all images reached the desired label with the desired confidence level ($p \geq 0.8$). The fast gradient method was significantly faster than the optimization one (and an iterative implementation increased its adversarial abilities from 98% to 100%) as shown in Table 1. Unfortunately, the median filter (of size 5x5) was able to remove the adversarial noise in 15% of the adversarial examples, hence the method is not secure enough.

3.4 IAN: Our New Approach to Adversarial Noise

In order to introduce a minimal perturbation to the original image, the fast gradient sign method [15] keeps the noise magnitude ε very small, thus compromising its robustness to removal attacks using filters. To resolve this problem, we suggest repeatedly applying the gradient sign method by gradually increasing the noise magnitude, until it cannot be removed by filtering. Hence, we aim to achieve an optimal trade-off between usability and security.

Our construction for the generation of immutable adversarial noise starts with an adversarial image, produced by the fast gradient sign method with a small noise constant ε. It then filters the adversarial image and tries to recognize it. If it succeeds then we increase the noise and iterate until the noise cannot be removed (the filtered image is recognized by the target network as the class of our choice). We detail the construction in the pseudocode shown in Algorithm 1.

A median filter of size 5x5 was used in our construction, as it experimentally showed the highest success in removing the adversarial noise generated by the fast gradient sign method when compared with other standard filters such as the average, Gaussian lowpass and Lapla-

[4] Different authors claim different security levels as the minimal standard for new CAPTCHA designs. In the literature we can find figures ranging from 0.6% to around 5%. Our security objective in this work is to propose a scheme that can only be successfully passed by bots 1% of the time, or less.

Method	Average Time	Std Time	Adversarial Success	Immutability Level
Optimization [30]	120.94s	98.19s	85.2%	16.2%
Our iterative implementation of Fast Gradient Sign [15]	0.81s	0.30s	100%	15%
Our method for IAN generation	1.01s	0.80s	100%	0%

Table 1: Comparison between adversarial noise generation methods. Reported times show the efficiency of the generation algorithm; Adversarial Success indicates the percentage of examples that succeeded to deceive the target DL network to classify the adversarial example with the chosen target category (chosen at random); Immutability level indicates the percentage of adversarial inputs (out of 1000) that were reverted to their original category by the means of median filter of size 5x5.

Algorithm 1 IAN_Generation

Require: *Net* a trained DL network; I a source image; C_i is the true class of I; C_d a deceiving class; p a confidence level of the network; M_f a Median filter.

Begin:

$adv(I, C_d, p) \leftarrow I$; $\quad \{adv(I, C_d, p)$ *the adversarial example*$\}$

$\Delta \leftarrow 0$;

while $Net(M_f(adv(I, C_d, p))) = C_i$ **do**

  while $Net(adv(I, C_d, p)) \neq C_d$ **or** confidence $< p$ **do**

    $\Delta \leftarrow$ run gradient sign method with noise magnitude ε;

    $adv(I, C_d, p) \leftarrow adv(I, C_d, p) + \Delta$;

  end while

  $\varepsilon = \varepsilon + \delta_\varepsilon$; $\quad \{$Increase the noise constant;$\}$

end while

Output: Δ

cian, and was faster than more complex filters such as non-local means [2] and wavelet denoising [19].

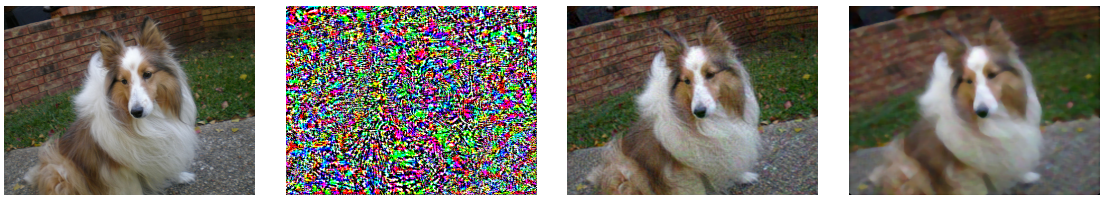
We tested the proposed method on the same set of 1000 images. The evaluation results, shown in Table 1, prove that our method for IAN generation satisfies all the requirements, as defined in Section 3.1. It is important to note that the additional checks to ensure robustness against the median filter M_f do not slow down the generation process significantly.

Figure 2c shows an example of an adversarial image, created by adding the IAN (Figure 2b), produced by our novel algorithm, to the original image (Figure 2a). Figure 2d depicts the result of applying the median filter to the adversarial image. The resulting image is not recognized correctly by a DL network. Moreover, the filtering moved the classification to a category, which is further away (in terms of the distance between the class positions in the score vector) from the true one. The distance between the true and deceived classes is 214, and between the true class and the class of the image after filtering (a removal attack) is 259. At the same time, while being more noticeable than in the previous algorithms, the relatively small amount of added noise still allows a human to easily recognize the image contents.

4 DeepCAPTCHA

We now propose a novel CAPTCHA scheme that we call DeepCAPTCHA, which is based on a large-scale recognition task, involving at least 1000 different categories. The scheme utilizes a DL network, trained to recognize these categories with high accuracy. DeepCAPTCHA presents an adversarial example as a recognition challenge. The adversarial example is obtained by creating and adding an IAN to its source image. The deceiving class in IAN must differ from the true class of the source image (both classes are from the categories, involved in the recognition task). The source image is chosen at random from a very large (bottomless) source of images with uniform distribution over classes and discarded once the adversarial image is created.

Contrarily to previous CAPTCHAs that use letters or digits, we use objects in order to make the classification task larger and to provide enough variability in the image to make it robust to attacks that aim to remove the adversarial noise. Using object recognition as a challenge



(a) (b) (c) (d)

Figure 2: An example of the IAN generation algorithm. Image (a) is the original image, correctly classified as a Shetland sheepdog with a high confidence of 0.6731 (b) is the computed immutable adversarial noise, (c) is the adversarial image (the sum of the image in (a) and the IAN in (b)), classified as a tandem or bicycle-built-for-two with a 0.9771 confidence and (d) the result of applying M_f, classified as a chainlink fence with confidence 0.1452.

poses two usability issues: 1) object names are sometimes ambiguous, 2) typing in the challenge solution requires adapting the system to user's language. We propose to solve these issues by providing a set of pictorial answers, i.e., a set of images, each representing a different class. Obviously, the answers contain the correct class, as well as random classes (excluding the deceiving class).

The task for the user is to choose (click on) the image from the supplied set of answers that belongs to the same class as the object in the test image – the adversarial example. Since we keep the adversarial noise small, a human could easily recognize the object in the adversarial example and choose the correct class as the answer. The only possible AI tool that can solve such a large-scale image recognition problem is a DL network. The adversarial noise used to create the adversarial example is designed to deceive the DL tools into recognizing the adversarial image as a different category. Hence, the proposed challenge is easy for humans and very difficult for automatic tools.

4.1 The Proposed Model

In this section we provide a formal description of our proposed design. Let *Net* be a DL network trained to classify n ($n \geq 1000$) classes with high (human-competitive) classification accuracy. Let $C = \{C_1, ..., C_n\}$ be a set of labels for these n classes. Let I be an image of class $C_i \in C$. Let $C_i^* = C \backslash \{C_i\}$,[5] and let C_d be a deceiving label which is chosen at random from C_i^*. The DeepCAPTCHA challenge comprises the following elements:

- An adversarial image $adv(I, C_d, p)$, constructed from I by the addition of an immutable adversarial noise component (constructed by Algorithm 1) that changes the classification by the DL *Net* to class C_d with confidence at least p.[6]
- $m-1$ answers, which can be fixed images corresponding to $m-1$ labels chosen at random and without repetition from $C_i^* \backslash \{C_d\}$;
- A fixed image with label C_i, different from I.

The $m-1$ suggestions and the true answer are shown in a random order. The challenge for the user is to choose the image from the answers that belongs to C_i. The original image I could be randomly picked from any number of databases and/or online social networks and other sources, and it is discarded after creating the adversarial example.

The pseudocode for the DeepCAPTCHA challenge generation is shown in Algorithm 2 and an example, generated by our proof-of-concept implementation (detailed in Section 6), is depicted in Figure 3.

5 Security Analysis

In the following we analyze several completely different but complementary approaches that potential attackers could use against the DeepCAPTCHA system. We start the analysis by discussing a straightforward guessing attack, we then continue to evaluate attacks that use image processing techniques, aiming to revert the adversarial image to its original class by applying image processing filters. Finally, we turn to more sophisticated attacks that employ machine learning tools. We set the security requirement for the success of an attack to 1%.

5.1 Random Guessing Attack

Using m answers per challenge provides a theoretical bound of $(\frac{1}{m})^n$ for the probability that a bot will successfully pass n challenges.[7] Therefore, $n = \frac{-\log p}{\log m}$ are

[5] We note that in some cases, depending on the variability of the data set, it is suggested to remove classes similar to C_i from C_i^*.

[6] In our experiments we have used $p = 0.8$.

[7] Assuming independence between tests.

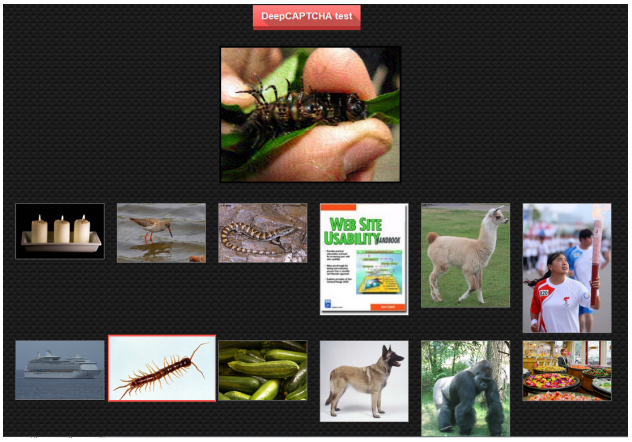


Figure 3: An example of a DeepCAPTCHA challenge. The large image above is the computed adversarial example, and the smaller ones are the set of possible answers.

Algorithm 2 Compute a DeepCAPTCHA challenge

Require: $[C_1, C_2, \ldots, C_n]$ a set of n classes; $\{I_j\}_{j=1}^{n}$ fixed representative images of the n classes (used as answers); $i \leftarrow_r [1, 2, \ldots, n]$ the index of a random class C_i, and $I \in_R C_i$ a random element; m the number of possible answers; p the desired misclassification confidence; Net a trained DL network; M_f a Median filter.

Begin:

Randomly pick a destination class $C_d, d \neq i$;

Set Δ = IAN_Generation(Net, I, C_i, C_d, p, M_f);

$adv(I, C_d, p) = I + \Delta$; {The immutable adversarial example}

Randomly select $m-1$ different indexes $j_1, \ldots, j_{m-1}$ from $[1, \ldots, n] \backslash \{i, d\}$;

Choose the representative images $[I_{j_1}, \ldots, I_{j_{m-1}}]$ of the corresponding classes;

Output: The CAPTCHA challenge is formed by the adversarial image $adv(I, C_d, p)$, and a random permutation of the set of m possible answers $\{I, I_{j_1}, \ldots, I_{j_{m-1}}\}$.

required for achieving a False Acceptance Rate (FAR)[8] of p. As we show later (in Section 6.1), $m = 12$ offers sufficient usability (low False Rejection Rate (FRR) and fast solution), hence for our target FAR of at most 1%, n should be greater than 1.85, e.g., $n = 2$ (resulting in an FAR of 0.7%).

One can alternatively combine challenges with different numbers of answers in consecutive rounds or increase n. These allow a better tailoring of the FAR or the FRR (both can be computed following the figures of Table 3). The latter approach offers a finer balance between security and usability.

5.2 Filtering Attacks

We examined the robustness of our IAN generating algorithm to a set of image filters particularly aimed to remove the noise addition. Any of these attacks will succeed if they are able to remove sufficient noise to correctly classify an adversarial example into the class of its original image.

Consequently, we tested seven filters with a wide range of parameters on a set of 1000 adversarial examples, created with the generation algorithm presented in Algorithm 1. This set of filters included the median filter, averaging filter, circular averaging filter, Gaussian low-pass filter, a filter approximating the shape of the two-dimensional Laplacian operator, non-local means [2], and wavelet denoising [19]. Table 2 shows the success rates of the different filters (along with the optimal parameter choice for the filter). The success rates of all filters are significantly below the security requirement of 1%.

5.3 Machine Learning Based Attacks

Two variants of machine learning attacks were applied to DeepCAPTCHA. Before this analysis, we define the attacker model:

[8] In our context, FAR stands for the probability that a bot succeeds to pass the DeepCAPTCHA whereas FRR stands for the probability that a human fails to pass the DeepCAPTCHA.

Median (Size 5x5)	Averaging (Size 5x5)	Circular Averaging (Radius 5)	Gaussian Lowpass (Size 5x5, std = 0.5)	Laplacian (Size 3x3, $\alpha = 0.2$)	Non-local Means (1/2 Patch size = 3, 1/2 Window size = 2, Weighting = 0.1)	Wavelet Denoising ($\sigma = 3$, Num. levels = 3)
0%	0%	0.1%	0.1%	0.2%	0.3%	0.2%

Table 2: Filters employed in the filter attack, and their respective success rates (out of 1000 trials). Note that the median filter was used in the generation process, thus the challenge is robust to the median filter by construction.

5.3.1 The Attacker Model

Knowledge of the algorithm and its internal parameters: The attacker has a full knowledge of the CNN (its architecture and parameters or knowledge of the training set that allows training of a similar CNN), used in the adversarial noise generation algorithm and of the generation algorithm itself, including its internal parameters.

Access to DeepCAPTCHA challenges: The attacker has access to all generated adversarial examples (but not to their source) as well as to the images which serve as the representatives of the classes (one or more per class).

No Access to the Source Images: The source images (used to generate the adversarial examples) are chosen at random from crawling a number of high volume online social media and similar sites, thus the size of the source image pool can be considered infinite for all practical purposes. Once the adversarial image is created, the corresponding original image is discarded instantly and never reused. Even though the attacker has theoretically equal access to all images that could be employed by the generator, he has no knowledge about the particular image used for generating the presented adversarial example.

Access to other machine learning tools: The attacker has the ability to use any other classifier in an attempt to classify the adversarial examples or to train the same or other DL networks on them. This is done with the aim of finding alternative networks with similar accuracy over the baseline classification problem, but having more robustness against adversarial examples.

Despite the above assumptions, that are very strong and extremely conservative, we are still able to develop a secure and usable DeepCAPTCHA.

5.3.2 Alternative Classifier Attack

The most straightforward attack on DeepCAPTCHA is probably the one that tries to use other classifiers, in an attempt to correctly recognize the adversarial example.

A machine learning algorithm, to be used successfully in such an attack, should be 1) robust to adversarial examples in general or at least to those used in the DeepCAPTCHA; 2) scalable to a large number of categories (+1000).

Highly non-linear models such as RBF SVM or RBF networks are known to be more robust to this adversarial phenomenon [11, 15], but these classifiers are not scalable to cover +1000 categories. Thus they do not present a practical method for breaking DeepCAPTCHA or future similar schemes.

Since the adversarial generation algorithm uses a specific network, one can consider a potential attack using another DL network with a different architecture and/or parameterization. However, it was previously shown that adversarial examples generalize well to different architectures and initializations [15, 30].

To verify the robustness of our construction against attacks that use alternative DL algorithms, we tested several publicly available DL networks on DeepCAPTCHA. Specifically, we used the CNN-F network from [6] to generate the CAPTCHA and we tested the ability to recognize the adversarial examples using three other deep learning networks. Two of these networks have a different architecture: CNN-M is similar to Zeiler and Fergus [38] and CNN-S is similar to OverFeat [27]. The third network – AlexNet from [17], has an architecture similar to CNN-F, with the difference that CNN-F has a reduced number of convolutional layers and a denser connectivity between convolutional layers. The CNN-M and CNN-S networks were only able to recognize correctly one out of the 1000 images in our test set, while AlexNet failed to classify any adversarial examples correctly. Consequently, none of these tools reached the 1% threshold.

5.3.3 Noise Approximation Attack

Given that the challenges were generated by adding adversarial noise, the attacker may hope to approximate this noise (to remove it) using DL. We show next that for suitably chosen image sources, this attack is successful less than 1% of the time. Recall that the images belong to known classes. Therefore, the attacker can try and explore the similarity between images of the same class in order to approximate the noise that changes the classification from the true category (C_i) to the deceiving one (C_d). Averaging or devising a "class" noise for all instances of the class does not seem practical, as the noise is subtle and averaging it over all images will most likely destroy it. A better idea is to collect representative samples of a category and learn a noise per each sample in that class and for each other category in the system.

For the attack to be effective, the adversarial noise that takes an element from C_i and "transforms" it into an element in C_d, should be relatively independent. This holds for classes with small intra-class variation, for example a category comprising images of the letter 'A' printed with a similar font.

For the attack to be effective, the variation between the instances of the same class should be small, for example a category comprising images of the letter 'A' printed with a similar font. In other words, the adversarial noise that takes an element from C_i and "transforms" it into an element in C_d, should be relatively independent of the actual element.

Fortunately, this property rarely holds for general objects categories like the ones we are using. In fact, this is what causes the baseline classification to be difficult in the first place, requiring a sophisticated feature extraction process (such as CNN) to overcome the very high intra-class variation.

Along these lines, we implemented and tested an attack we have named the *noise approximation attack*. Consider a working example with the following settings: a thousand image categories, where each category is represented by 1200 images[9] and the CAPTCHA is set to 12 answers.

If the images used for answers are static, then their labels could be pre-computed by running the network over all classes only once. Then, for each challenge, the labels of the answers could be retrieved very efficiently.

In the pre-computation step, the attacker can compute the adversarial noises that transform every image in the dataset into every other category. This implies a total of $1,200 * 999 = 1,198,800$ adversarial noises (i.e., a representative image $I' \in C_i$ and a target category d compute all its $\Delta^d_{I',i} = I' - adv(I, C_d, p)$ values).

In the online phase of the attack, the attacker is presented with the challenge, including the adversarial example[10] $adv(I, C_d, p)$ and a random permutation of 12 possible answers $\{I_i, I_{j_1}, \ldots, I_{j_{11}}\}$ (where i is the label of the correct class, and d is the decoy label of the adversarial example). Then, the attacker runs the network over $adv(I, C_d, p)$ and retrieves the decoy label d. As the attacker knows that the noise caused the image I to be classified in C_d (rather than one of the 12 classes represented by the set of answers), he tries to remove the adversarial noise that transforms $I' \in C_i$ into C_d from $adv(I, C_d, p)$. Specifically, for each class j of the 12 answers, and for each representative image $I' \in C_j$, the attacker computes the estimation of the original image as: $I^* = adv(I, C_d, p) - \Delta^j_{I',i}$, and then runs the network on the estimate I^*, which results in $1200 * 12 = 14400$ attempts per challenge (as the representative sets are of size 1200 images, and there are 12 candidate sets). This is a large number, but if the images in the same category are very similar (e.g., same digit), then even the first attempt could be successful. To prevent such security issues one should exclusively use natural images of real objects with moderate to high intra-class variation as a source for CAPTCHA generation.

We ran an instance of the noise approximation attack, where the true category was *lion* (that exhibits moderate intra-class variation) and the target category was *rooster*. A total of 3 out of 1200 challenges were broken using this approach. This implies that the noise approximation attack is interesting and relevant, and despite its low success rate of 0.25%, needs to be taken into account and considered in future implementations to ensure it stays below the 1% threshold.

6 PoC: DeepCAPTCHA-ILSVRC-2012 System

We implemented a proof-of-concept system using CNN-F deep network from [6], trained on ILSVRC-2012 database [26]. This set contains 1000 categories of natural images from ImageNet. The DL network was trained on the training set of the ILSVRC-2012 database, and we used the validation set that contains 50,000 images as a pool for source images. For each challenge we picked an image at random and produced an adversarial example for it using the IAN generation method, detailed in Algorithm 1. We selected one representative image per category from the training set (to guarantee that the answers do not contain the image, used to generate adversarial examples) for the answers.

[9] To make the CAPTCHA more secure, we chose classes with large variability between the categories.

[10] We remind the reader that I is **not** available to the adversary as per our assumptions.

The PoC system was implemented as a web application in order to conduct a usability test. In our implementation we varied the number of answers to test the best trade-off between usability and security (more choices increase the security, but are harder for users and the solution takes more time). The number of challenges per session was set to 10, in order to run the usability statistics (note that our security analysis suggests that 2–3 answers are enough to reach the desired security level). An example of a challenge from the PoC system is shown in Figure 3.

6.1 Usability Analysis of the PoC System

We tested a proof-of-concept implementation of our DeepCAPTCHA using 472 participants contacted using the Microworkers.com online micro crowd sourcing service. Each participant was requested to provide anonymous statistical data about their age, gender and familiarity with computers before starting the test. Participants were next presented with 10 DeepCAPTCHA challenges of varying difficulties and gave feedback on usability once they had completed the challenges. This provided us with 4720 answered tests, of which we removed 182 (approx. a 3,85%) to avoid outliers. In particular, we removed tests or sessions if they fall into any of these three categories[11]: 1. Sessions with average time per test higher than 40 seconds, 2. Tests with answer times above 45 seconds, and 3. Sessions with a success rate of 10% or lower.

We tried to get some insights into the best trade-off between usability and security by testing different numbers of answers, in the range $8+4k, k \in \{0,\ldots,3\}$, so users were randomly assigned variants of the tests with different number of answers for studying the impact of this change. The most relevant usability results are shown in Table 3. The participants reported high satisfaction with DeepCAPTCHA usability (see Figure 4). The data shown in Figure 4 is an average across all variants, from 8 to 20 answers. As expected, the perceived user-friendliness and difficulty (see Figure 5) of the CAPTCHA deteriorated steadily from the versions with 8 answers to those with 20.

It is interesting to note that participants who declared their gender as female performed significantly better than the males, across all variants, the gap becoming wider with the increasing difficulty of the CAPTCHA task, as seen in Figure 6. Consistent with this finding is the additional fact that females not only achieved better accuracy but also did it using less time on average than males.

[11] We assume that high solving times are due to users that were interrupted during the tests, and the low success rates are due to users that did not follow the instructions, or chose their answers at random.

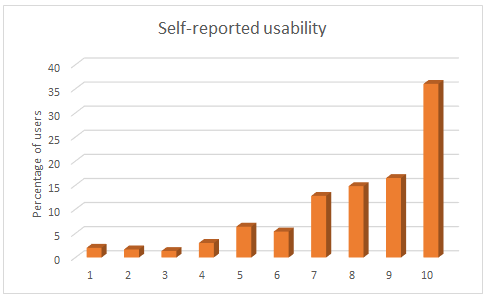


Figure 4: Self reported user friendliness of DeepCAPTCHA. Answers in the range 1-10, 10 being best.

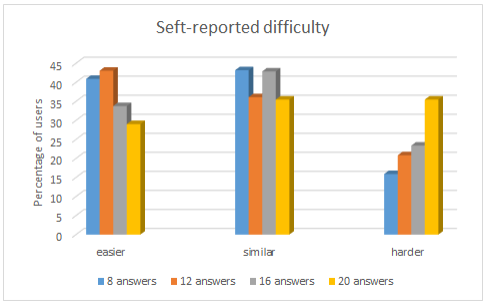


Figure 5: Self reported DeepCAPTCHA difficulty, compared with existing CAPTCHAS, for variants from 8 to 20 answers.

We define a secure CAPTCHA as one that has a less than 1% chance of being successfully attacked by a bot, and a usable CAPTCHA as one with a challenge pass rate above 75% when attempted by a human within an average time of 15s. These thresholds are in line with those previously reported and with other CAPTCHA schemes.

Based on the results collected so far in our preliminary tests, and the security analysis in Section 5, we conclude that the best trade-off between security and usability is met by the version of our test with 12 answers per challenge and two challenges in a CAPTCHA session. This configuration meets the accepted security and usability requisites. Namely, humans showed a success rate of 86.67% per challenge, hence the overall success probability is (assuming independence) about $0.8667^2 = 0.751$. The average time for the session was about $2 \cdot 7.66s = 15.32s$ (the median is significantly faster — $10.4s$). The security analysis showed that a probability of a bot bypassing the scheme is not higher than 0.7% (by random guessing).

We expect that users will become more familiar with the task and the system in general as it gains popularity. This would result in higher success rates and faster solution times.

	Overall Results	8 answers	12 answers	16 answers	20 answers
Total test count	4538	1257	990	1144	1147
Success rate	82,57 %	89,18 %	86,67 %	79,98 %	74,37 %
Average time	7,89s	6,04s	7,66s	8,36s	9,66s
Median time	5,49s	4,24s	5,18s	5,89s	7,34s

Table 3: Usability results for the DeepCAPTCHA proof of concept implementation, with different number of answers.

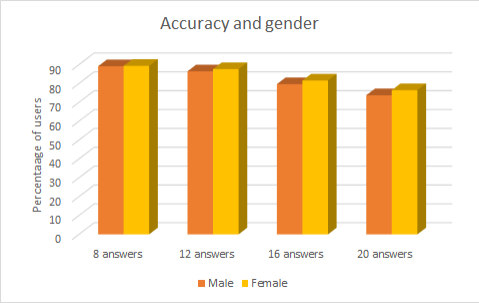


Figure 6: Accuracy across self-reported gender for variants from 8 to 20 answers.

6.2 Discussion of the PoC System

The best candidates for image source in DeepCAPTCHA are natural images of objects. Such images have large enough variation in color and intra-class variability. Another important property (related to security) that exists in such images is that they belong to a large variety of classes, allowing to run a recognition with 1000 or more categories.

We based the PoC system on the ILSVRC-2012 database [26] which constitutes the best available option at the moment. However, this database is suboptimal, as it contains many categories that are culturally influenced (i.e., bird species only existent in certain parts of the world) or similar to each other (i.e., different breeds of dog that are almost identical). These shortcomings certainly have an effect on usability, security, and scalability of the system, which we specify in the following discussion.

Usability: Unfamiliar or too similar objects make human recognition harder. Comparing the test image to a relatively small set of answers partially resolves the problem, as the probability of similar categories to appear in the set of possible answers is not very high, though it is still non zero.

Another usability problem is related to images depicting complex scenes containing several objects. There is a possibility that more than one of the displayed objects will appear among the answers, making the test ambiguous. This can be solved by analyzing the top recognition scores of the network for the original image and removing the categories corresponding to the top scores from the pool of answers (not implemented in PoC system).

Scalability: Due to the similarity between some of the classes in ILSVRC-2012, the state-of-the-art networks show only about 60% top 1 rank classification on the original images from the validation set. The top 5 rank classification accuracy is significantly higher, reaching 81-85% (depending on the architecture).

A low top 1 rank classification accuracy introduces scalability problems. As we require to produce a large number of CAPTCHA challenges per second, we need an automatic way of obtaining the labels. A 40% misclassification rate makes it impossible to fully automate the process, as the true label of the source image I is required by our algorithm. Merging similar classes is not viable as it decreases the number of classes, thereby directly reducing the underlying problem difficulty and, hence, security.

The average time for generating IAN is reported for a Matlab implementation (including CNN). We expect it to improve significantly in more optimized environment.

Security: A low top 1 rank classification accuracy adversely affects the security of the system. Our empirical study showed that 2.4% of the adversarial examples, created from the validation set of ILSVRC-2012, include the true class in their top 5 rank classification (but not the top 1). Moreover, 21.6% of the filtered adversarial images (with a median filter) include the true class in their top 5 rank classification. A simple attack would then be to look at the intersection between the top 5 classification of the adversarial examples in the CAPTCHA and the set of its answers. In the case where the intersection contains a single class, the true label has been determined. This problem arises only when the network fails to classify the original image correctly in the top 1 rank and the true class appears in the next 2–5 scores.

Note that improving the network classification accuracy will improve both scalability (for automatic labeling) and security (a scoring function that has a sharp peak at the true category will eliminate the security problem observed for the top 5 rank in the ILSVRC-2012 data set). We stress that the issues discussed above are strictly linked to the data set, not to our proposal. For deploying

a real system, one should consider collecting a data set with 1000 or more categories, with high inter class variability, discarding complex multi-object scenes, in particular those including a number of objects featured in the database.

7 Conclusions and Future Work

In this work, we introduced DeepCAPTCHA, a secure new CAPTCHA mechanism based on immutable adversarial noise that deceives DL tools and cannot be removed using preprocessing. DeepCAPTCHA offers a playful and friendly interface for performing one of the most loathed Internet-related tasks — solving CAPTCHAs. We also implemented a first proof-of-concept system and examined it in great detail.[12]

We are the first to pose the question of adversarial examples' immutability, in particular to techniques that attempt to remove the adversarial noise. Our analysis showed that previous methods are not robust to such attacks. To this end, we proposed a new construction for generating immutable adversarial examples which is significantly more robust to attacks attempting to remove this noise than existing methods.

There are two orthogonal directions for future CAPTCHA research 1) Design a new large-scale classification task for DeepCAPTCHA that contains a new data set of at least 1000 dissimilar categories of objects. This task also includes collecting (and labelling) a new data set for training of the CNN. 2) Introduction of CAPTCHAs based on different modalities, such as sound/speech processing (e.g., to address users with visual impairments).

Finally, we believe that IAN has a wide range of applications in computer security. IANs may be used to bypass current ML-based security mechanisms such as spam filters and behavior based anti-malware tools. Additionally, our proposed attacks on adversarial noise may be of independent interest.

8 Acknowledgments

The authors thank Daniel Osadchy for his worthy contributions to the paper. The funds received under the binational UK Engineering and Physical Sciences Research Council project EP/M013375/1 and Israeli Ministry of Science and Technology project 3-11858, "Improving cyber security using realistic synthetic face generation" allowed this work to be carried out.

[12] DeepCAPTCHA can be accessed at `http://crypto.cs.haifa.ac.il/~daniel`

References

[1] ARE YOU A HUMAN. `http://www.areyouahuman.com/`, Last accessed January 2016.

[2] BUADES, A., COLL, B., AND MOREL, J. A Non-Local Algorithm for Image Denoising. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005), 20-26 June 2005, San Diego, CA, USA* (2005), pp. 60–65.

[3] BURSZTEIN, E. How we broke the NuCaptcha video scheme and what we proposed to fix it. `http://elie.im/blog/security/how-we-broke-the-nucaptcha-\video-scheme-and-what-we-propose-to-\fix-it/` Last accessed January 2016.

[4] BURSZTEIN, E., AIGRAIN, J., MOSCICKI, A., AND MITCHELL, J. C. The End is Nigh: Generic Solving of Text-based CAPTCHAs. In *Proceedings of the 8th USENIX Conference on Offensive Technologies* (Berkeley, CA, USA, 2014), WOOT'14, USENIX Association, pp. 3–3.

[5] BURSZTEIN, E., BETHARD, S., FABRY, C., MITCHELL, J. C., AND JURAFSKY, D. How Good Are Humans at Solving CAPTCHAs? A Large Scale Evaluation. In *Proceedings of the 2010 IEEE Symposium on Security and Privacy* (Washington, DC, USA, 2010), SP '10, IEEE Computer Society, pp. 399–413.

[6] CHATFIELD, K., SIMONYAN, K., VEDALDI, A., AND ZISSERMAN, A. Return of the Devil in the Details: Delving Deep into Convolutional Nets. In *British Machine Vision Conference, BMVC 2014, Nottingham, UK, September 1-5, 2014* (2014).

[7] CHELLAPILLA, K., LARSON, K., SIMARD, P., AND CZERWINSKI, M. Designing Human Friendly Human Interaction Proofs (HIPs). In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2005), CHI '05, ACM, pp. 711–720.

[8] DAHL, G. E., SAINATH, T. N., AND HINTON, G. E. Improving deep neural networks for LVCSR using rectified linear units and dropout. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2013, Vancouver, BC, Canada, May 26-31, 2013* (2013), pp. 8609–8613.

[9] DATTA, R., LI, J., AND WANG, J. Z. IMAGINATION: a robust image-based CAPTCHA generation system. In *Proceedings of the 13th ACM International Conference on Multimedia, Singapore, November 6-11, 2005* (2005), pp. 331–334.

[10] ELSON, J., DOUCEUR, J. R., HOWELL, J., AND SAUL, J. Asirra: A CAPTCHA that Exploits Interest-Aligned Manual Image Categorization. In *Proceedings of 14th ACM Conference on Computer and Communications Security (CCS)* (October 2007), Association for Computing Machinery, Inc.

[11] FAWZI, A., FAWZI, O., AND FROSSARD, P. Analysis of classifiers' robustness to adversarial perturbations. *CoRR abs/1502.02590* (2015).

[12] GIRSHICK, R. B., DONAHUE, J., DARRELL, T., AND MALIK, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014* (2014), pp. 580–587.

[13] GOLLE, P. Machine Learning Attacks Against the Asirra CAPTCHA. In *Proceedings of the 15th ACM Conference on Computer and Communications Security* (New York, NY, USA, 2008), CCS '08, ACM, pp. 535–542.

[14] GOODFELLOW, I. J., BULATOV, Y., IBARZ, J., ARNOUD, S., AND SHET, V. D. Multi-digit number recognition from street view imagery using deep convolutional neural networks. *CoRR abs/1312.6082* (2013).

[15] GOODFELLOW, I. J., SHLENS, J., AND SZEGEDY, C. Explaining and harnessing adversarial examples. *CoRR abs/1412.6572* (2014).

[16] HERNÁNDEZ-CASTRO, C. J., RODRÍGUEZ-MORENO, M. D., AND BARRERO, D. F. Using JPEG to measure image continuity and break capy and other puzzle captchas. *IEEE Internet Computing 19*, 6 (2015), 46–53.

[17] KRIZHEVSKY, A., SUTSKEVER, I., AND HINTON, G. E. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States.* (2012), pp. 1106–1114.

[18] LECUN, Y., BOSER, B. E., DENKER, J. S., HENDERSON, D., HOWARD, R. E., HUBBARD, W. E., AND JACKEL, L. D. Backpropagation applied to handwritten zip code recognition. *Neural Computation 1*, 4 (1989), 541–551.

[19] MIHÇAK, M. K., KOZINTSEV, I., AND RAMCHANDRAN, K. Spatially adaptive statistical modeling of wavelet image coefficients and its application to denoising. In *Proceedings of the 1999 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP '99, Phoenix, Arizona, USA, March 15-19, 1999* (1999), pp. 3253–3256.

[20] MNIH, V., KAVUKCUOGLU, K., SILVER, D., RUSU, A. A., VENESS, J., BELLEMARE, M. G., GRAVES, A., RIEDMILLER, M., FIDJELAND, A. K., OSTROVSKI, G., PETERSEN, S., BEATTIE, C., SADIK, A., ANTONOGLOU, I., KING, H., KUMARAN, D., WIERSTRA, D., LEGG, S., AND HASSABIS, D. Human-level control through deep reinforcement learning. *Nature 518*, 7540 (Feb. 2015), 529–533.

[21] MOHAMED, M., SACHDEVA, N., GEORGESCU, M., GAO, S., SAXENA, N., ZHANG, C., KUMARAGURU, P., VAN OORSCHOT, P. C., AND CHEN, W. A three-way investigation of a game-CAPTCHA: automated attacks, relay attacks and usability. In *9th ACM Symposium on Information, Computer and Communications Security, ASIA CCS '14, Kyoto, Japan - June 03 - 06, 2014* (2014), pp. 195–206.

[22] NAOR, M. Verification of a human in the loop or Identification via the Turing Test. `http://www.wisdom.weizmann.ac.il/~naor/PAPERS/human_abs.html`.

[23] NGUYEN, A. M., YOSINSKI, J., AND CLUNE, J. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015* (2015), pp. 427–436.

[24] NUCAPTCHA. Whitepaper: NuCaptcha & Traditional Captcha. `http://download.nudatasecurity.com/nucaptcha-vs-traditional-captcha/`, Last accessed January 2016.

[25] PAPERNOT, N., MCDANIEL, P., JHA, S., FREDRIKSON, M., CELIK, Z. B., AND SWAMI, A. The limitations of deep learning in adversarial settings. *CoRR abs/1511.07528* (2015).

[26] RUSSAKOVSKY, O., DENG, J., SU, H., KRAUSE, J., SATHEESH, S., MA, S., HUANG, Z., KARPATHY, A., KHOSLA, A., BERNSTEIN, M. S., BERG, A. C., AND LI, F. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision 115*, 3 (2015), 211–252.

[27] SERMANET, P., EIGEN, D., ZHANG, X., MATHIEU, M., FERGUS, R., AND LECUN, Y. Overfeat: Integrated recognition, localization and detection using convolutional networks. In *International Conference on Learning Representations (ICLR 2014)* (April 2014), CBLS.

[28] SUPHANNEE SIVAKORN, IASONAS POLAKIS, A. D. K. I am robot:(deep) learning to break semantic image captchas. In *1st IEEE European Symposium on Security and Privacy* (2016).

[29] SZEGEDY, C., LIU, W., JIA, Y., SERMANET, P., REED, S., ANGUELOV, D., ERHAN, D., VANHOUCKE, V., AND RABINOVICH, A. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015* (2015), pp. 1–9.

[30] SZEGEDY, C., ZAREMBA, W., SUTSKEVER, I., BRUNA, J., ERHAN, D., GOODFELLOW, I. J., AND FERGUS, R. Intriguing properties of neural networks. *CoRR abs/1312.6199* (2013).

[31] TAIGMAN, Y., YANG, M., RANZATO, M., AND WOLF, L. Web-scale training for face identification. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015* (2015), pp. 2746–2754.

[32] VEDALDI, A., AND LENC, K. MatConvNet: Convolutional Neural Networks for MATLAB. In *Proceedings of the 23rd Annual ACM Conference on Multimedia Conference, MM '15, Brisbane, Australia, October 26 - 30, 2015* (2015), pp. 689–692.

[33] VON AHN, L., BLUM, M., AND LANGFORD, J. Telling Humans and Computers Apart Automatically. *Commun. ACM 47*, 2 (Feb. 2004), 56–60.

[34] VON AHN, L., MAURER, B., MCMILLEN, C., ABRAHAM, D., AND BLUM, M. reCAPTCHA: Human-Based Character Recognition via Web Security Measures. *Science 321*, 5895 (August 2008), 1465–1468.

[35] XU, Y., REYNAGA, G., CHIASSON, S., FRAHM, J., MONROSE, F., AND VAN OORSCHOT, P. C. Security analysis and related usability of motion-based captchas: Decoding codewords in motion. *IEEE Trans. Dependable Sec. Comput. 11*, 5 (2014), 480–493.

[36] YAN, J., AND AHMAD, A. S. E. Breaking Visual CAPTCHAs with Naive Pattern Recognition Algorithms. In *23rd Annual Computer Security Applications Conference (ACSAC 2007), December 10-14, 2007, Miami Beach, Florida, USA* (2007), pp. 279–291.

[37] YAN, J., AND EL AHMAD, A. S. Usability of CAPTCHAs or Usability Issues in CAPTCHA Design. In *Proceedings of the 4th Symposium on Usable Privacy and Security* (New York, NY, USA, 2008), SOUPS '08, ACM, pp. 44–52.

[38] ZEILER, M. D., AND FERGUS, R. Visualizing and understanding convolutional networks. In *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I* (2014), pp. 818–833.

[39] ZHU, B. B., YAN, J., LI, Q., YANG, C., LIU, J., XU, N., YI, M., AND CAI, K. Attacks and design of image recognition CAPTCHAs. In *Proceedings of the 17th ACM Conference on Computer and Communications Security, CCS 2010, Chicago, Illinois, USA, October 4-8, 2010* (2010), pp. 187–200.