A Honey Turing Test

Jeff Yan

University of Southampton, UK
`jeff.yan@soton.ac.uk`

In loving memory of Ross Anderson
— a relentless trailblazer, an inspiring mentor, and a cherished friend.

Abstract. How shall we distinguish computers from humans once machines can pass the Turing test? We explore the concept of a Honey Turing test, in which machines are given trap questions designed to reveal their identities. Importantly, this test must also ensure that humans are not mistakenly classified as computers. The judge in this setting can be either a human or a machine.

1 Introduction

With the rapid advancement of AI technologies, the Turing test [9] may fail to distinguish between robots and humans, sooner or later. In a future world, robots might be virtually identical to humans. For example, humanoid robots may behave and converse like human beings, expressing and reacting to emotions just as humans do. As a result, the Voight-Kampff test – an empathy test conceived in the sci-fi work *Blade Runners* – or similar concepts would no longer be effective. In this future world, how shall we tell which is a robot and which is a human?

We ask this question for three purposes: 1) exploring how one will differentiate between humans and machines, as a matter of general curiosity; 2) investigating its potential connections to and applications in cyber security; and 3) pondering its ramifications in psychology, human-computer interaction and future social norms.

2 Technical backgrounds

The Turing test [9] involves three participants: a human interrogator, a human respondent, and a machine. The interrogator communicates with the other two through a text interface and must determine which is the human and which is the machine based on their responses. If the machine can imitate human responses well enough that the interrogator cannot reliably distinguish it from the human, then the machine is said to have passed the Turing Test.

Turing argued that the question “Can machines think?” was ambiguous and suggested re-framing it as whether a machine can successfully perform in the imitation game. By focusing on the observable behaviour of computers, Turing provided an operational definition of intelligence — one that can be practically tested.

An automated Turing test, proposed by von Ahn et al [10], aims to distinguish humans and computers automatically. Their key insight was to leverage hard AI problems that computers cannot solve but which humans can. Various designs known as Captchas have been widely deployed on the Internet. Bots are supposedly recognised if they fail the Captcha tests, whereas humans are recognized as humans only if they pass the tests. In the same vein, additional security primitives [13] were developed using hard AI problems to achieve entity authentication and bot defense simultaneously.

The deep learning revolution has significantly reduced the cognitive gap between computers and humans, particularly in text, image and voice recognition. As a result, the design space for creating effective and user-friendly Captchas has become more limited.

Other methods. Something that machines can perform but humans cannot (e.g. transcribing sounds inaudible to humans) could be used to achieve a certain level of distinguishability between machines and humans, but this approach has its limitations. For example, the inability to answer a given question in this context does not necessarily indicate that the entity is human. A dumb robot will not be able to give a proper answer, either.

3 A conceptual sketch

A honey Turing test (HTT) involves three participants: one judge — who can be a human, a computer or a human assisted by a computer — a human, and a robot. The judge's goal is to accurately distinguish between the human and the robot without misclassification.

The judge may give each participant a number of challenges. Each challenge conceptually includes two parts: a honey (part which is presumably perceivable by robots only, and a plain part which is perceivable by humans only. Given the same challenge, a robot will return an answer that is predictable, definitive but unlikely to be produced by any humans, while a human will provide a different answer that can be easily verified as a human answer.

Machines should only be able to solve the honey part, and humans only solve the plain part. The answers from humans and robots should differ significantly and in a deterministic way. For example, in text recognition tasks, if a machine's responses resemble human typos – mostly correct characters with one error or two – the tests will fail to tell humans and bots apart.

In one HTT incarnation, a challenge is a combination of a honey part and a plain part, similar to a quantum bit (qubit), which exists in two states simultaneously. In another incarnation, we may have honey (or trap) challenges that are entirely separate from plain (or normal) ones, and the judge serves them alternately or in a random order.

What does it mean for a bot to pass or fail an HTT? If a bot passes the test, the judge has failed to identify it as a bot. If it fails the test, the judge has correctly identified it as a bot.

4 Assumptions and properties

We clarify what will be reasonable assumptions for such a scenario, and we discuss the key properties needed to make the test effective.

Honesty — or more precisely, truthfulness — is widely considered a necessary (though insufficient) condition for the safety of superintelligent systems; see, for example, the research agenda [1] of the AI Security Institute in the UK.

In contract, we consider the case where a bot is capable of deception, for three simple reasons: 1) if a bot is truthful, it will honestly reveal its identity when questioned; 2) in cyber security scenarios, bots are typically deceptive; and 3) it would be naive to assume that a superintelligent systems controlled by an adversary, such as an enemy state or a terrorist organisation, is honest and not deceptive or malicious. As a side note, in formulating his imitation game, Turing himself suggested that a machine might deliberately disguise its superhuman abilities (e.g., in calculation) to avoid revealing its identity too easily [9].

A robot should not be able to distinguish between a honey challenge and a plain one. This indistinguishability appears to be an essential property for an HTT to work; otherwise, when facing a honey challenge, a bot could choose to respond strategically, for example, by remaining silent to avoid being recognized.

5 Possible constructs and a simple attack

Adversarial examples [8] in neural networks exhibit intriguing properties: tiny, often imperceptible changes in input can mislead the classifier into producing an entirely different result. For example, after applying tiny perturbations to a cat image, a neural network may no longer recognise it as a cat but instead classify it as guacamole. However, humans can still easily recognise the perturbed image as a cat.

Adversarial examples offer promising potential for designing a honey Turing test. The key is to create adversarially perturbed images as challenges, where a machine will produce answers that are impossible for any human to generate and that differ from human responses. In other words, the machine perceives only the 'honey' part of the challenge, while the human perceives only the plain part.

Shi et al [6] proposed advCAPTCHA, which applied adversarial machine learning to improve the robustness of text Captchas. They embedded carefully crafted adversarial perturbations into normal text Captcha images to disrupt deep-learning-based attacks. However, the only application scenario they considered was deploying their design as a new Captcha scheme.

Here, I propose a simple yet broadly applicable attack to show that advCAPTCHA and similar designs will fail as an effective Honey Turing Test.

5.1 A distinguishing attack

There are at least three approaches to solving Captchas automatically:

– attacks that use deep learning models,
– solvers that rely on other machine learning methods rather than deep learning,
– methods that exploit security vulnerabilities without using either machine learning or deep learning (e.g., those in [2,12]).

The first two approaches both have general-purpose solvers capable of solving a wide range of designs; the third does not have a single algorithm that works universally. The last two approaches may also be immune to adversarial examples.

In theory, this offers a simple way for a bot to determine whether a challenge is a trap:

– solve the challenge using both a DL-based method and another method (one with a higher success rate than the third);
– if both methods produce the same result, the challenge is considered not a trap;
– if the results differ, it may or may not be a trap, and the bot can conceal its identity by outputting the answer produced by the non-DL method.

In this way, there is a good chance the machine can tell which challenge is a trap and which is likely not. It can also determine which answer is intended for the trap and which one is meant to appear human.

5.2 From theory to reality

The classic work by Daugman [3] first suggested that simple cells in the visual cortex of mammalian brains can be modelled using Gabor functions. In other words, human perception is similar to image analysis with Gabor filters. Based on these neuroscience-inspired insights, we reported in NDSS'16 a generic algorithm capable of solving a wide range of complicated text Captchas [4], including those far more complicated than the ones used in [6]. It has three major components: a Log-Gabor filter for directional decoding, a graph search algorithm for encoding, and a k-nearest neighbours (KNN) recognizer as the recognition engine. In essence, our approach computationally approximates the human process of Captcha solving in a reliable way.

Three main features make the algorithm in [4] a strong candidate for turning the distinguishing attack described above from theory into practice.

First, because the algorithm approximates human perception, it is inherently more robust to imperceptible adversarial examples than other approaches. If a perturbed challenge can be solved by humans, the algorithm is also highly likely to succeed.

Second, the algorithm does not use deep learning; instead, it relies on KNN, a non-DL method. As a simple distance-based method, KNN does not suffer from gradient-based attacks, whereas adversarial attacks on deep learning models typically rely on gradients to craft perturbations. Moreover, all available evidence in the literature—both theoretical and empirical—supports KNN's superior robustness to adversarial examples. For example, Papernot et al. [5] empirically demonstrated that adversarial examples created for deep neural network have poor transferability to KNN. Black-box attacks using substitute classifiers were successful for enabling cross-technique adversarial attacks. However, attacks employing other types of substitute classifiers have proven ineffective against nearest neighbor methods [5,11]. Generating KNN-specific adversarial examples remains an active research area, with recent progress reported in Wang et al [11] and Sitawarin et al [7].

Third, the algorithm has proven to be an effective and general solver.

5.3 Other devils in the details

As a side note, some other weaknesses in advCAPTCHA [6] warrant further significant research for improvement.

First, their design was not very secure even against DL-based solvers—the team's own target—and bots achieved a success rate of 41% to 46%. Large-scale fine-tuning reduced the bots' average success rate from 43% to 23% in a laboratory setting, but it is unclear how this will translate to the real world.

Second, its design clearly had usability issues. Their perturbation method resulted in many single-letter recognition errors by machines, resembling human typos. These had serious security implications, as defenders could not reliably distinguish between bots and humans. As a result, they had to rely on a complex process and human expert involvement in decision-making. We note that, in the context of Honey Turing Tests, human-like typo errors could make it very difficult to distinguish between bots and dyslexic users.

6 Lessons

Improving Captcha robustness against deep-learning attacks using adversarial examples is a *different* research problem from designing a viable Honey Turing Test. The distinction becomes clear when viewed through the lens of machine intelligence pertinent to each problem.

It may be arguably acceptable, in the former case, to assume that a robot uses only deep learning techniques and always provides a direct answer to any question it is given.

However, this would be an overly simplistic model of machine intelligence for designing a Honey Turing Test, which requires a more sophisticated and cunning intelligence model for machines — ultimately encompassing Artificial General Intelligence (AGI).

7 Concluding remarks

For a bot to pass a Honey Turing Test, it is necessary but insufficient to distinguish between trap challenges and normal ones. In addition, when faced with a trap challenge, the bot must respond in a human-like way to avoid being identified as a bot.

Current designs like adversarial text captchas will not work as an effective Honey Turing Test, since a robot can pass the test with a high probability using our simple attack.

Designing a Honey Turing Test using adversarial examples remains an open problem, as any viable approach must withstand both DL-based solvers and other types of solvers. To date, no design has successfully demonstrated such resistance. Achieving indistinguishability between trap and normal challenges is non-trivial. Moreover, while the rationale for using adversarial examples may be sound and promising, the devil is in the details. A deeper challenge lies in designing a Honey Turing Test for a world where machine learning becomes robust to adversarial examples.

It is also worth asking whether AGI will eventually make it impossible to distinguish robots from humans. Imagine a robot equipped with dual perception: one channel powered by cutting-edge machine vision, and another engineered to mechanically emulate human perception. At the same time, an AGI would likely master the entirety of human knowledge and thus become highly skilled at detecting any known traps designed to expose it.

We believe there is still good reason to be hopeful that new designs of Honey Turing Tests are possible. Our reasoning is as follows.

Adversarial examples use crafted inputs that exploit vulnerabilities in a model's decision boundary to cause misclassification. These attacks operate within the intended input-output interface of the model, not by observing unintended leaks. So, adversarial examples are typically viewed as input manipulation attacks, not side-channel attacks.

However, a key insight is the following. The use of adversarial examples as the honey part, combined with a plain part, essentially exposes the inner workings of an algorithm. If the answers a judge receives correspond to the honey parts consistently or more often than the plain parts, the judge can infer that the algorithm is DL-based; if they correspond to the plain parts consistently or more often, the judge can infer that the algorithm is not DL-based. By observing the relationship between inputs and outputs, some information about the internals of the algorithm can be deterministically inferred. In this sense, it constitutes an effective side channel.

We note that this discussion does not consider the distinguishing attack we proposed, and how to address this attack remains an open problem. However, we believe that in the era of AGI, side channels—whether through adversarial examples or other methods—will play a crucial role in identifying deceptive bots, thereby contributing to the development of effective Honey Turing Tests.

In this future world, humanoid robots will likely cohabit on this planet as if they were a living species. How will people react to that? How will people interact with them? How will social norms evolve accordingly? These are fascinating but still open problems. The only thing certain is that an exciting new world lies ahead of us!

Acknowledgement

I am grateful to Aurélien Bourquard and Ping Wang for valuable discussions.

References

1. AI Security Institute, Our Research Agenda, `https://www.aisi.gov.uk/research-agenda`, May 2025.
2. Algwil, A., Yan, J.: Failures of Security APIs: A New Case. Financial Cryptography and Data Security 2016: 283-298
3. J Daugman. Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters. JOSA A, 2(7):1160–1169, 1985.
4. Haichang Gao, Jeff Yan, et al. A Simple Generic Attack on Text Captchas. NDSS 2016.
5. Nicolas Papernot, Patrick McDaniel, Ian Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. arXiv preprint arXiv:1605.07277. 2016.
6. C. Shi, S. Ji, Q. Liu, C. Liu, Y. Chen, Y. He, Z. Liu, R. Beyah, and T. Wang, "Text captcha is dead? a large scale deployment and empirical study," ACM CCS, 2020
7. Chawin Sitawarin, Evgenios Kornaropoulos, Dawn Song, David Wagner. Adversarial Examples for k-Nearest Neighbor Classifiers Based on Higher-Order Voronoi Diagrams. NeurIPS 2021.
8. Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. arXiv:1312.6199, 2013.
9. A. M. Turing, "Computing machinery and intelligence," Mind, vol. 59, no. 236, pp. 433–460, 1950. Available: `http://www.jstor.org/stable/2251299`
10. Luis von Ahn, Manuel Blum, Nicholas J. Hopper, John Langford: CAPTCHA: Using Hard AI Problems for Security. EUROCRYPT 2003: 294-311
11. Y. Wang, S. Jha, and K. Chaudhuri. Analyzing the robustness of nearest neighbors to adversarial examples. Proc. of the 35th International Conference on Machine Learning, 2018. PMLR. URL http://proceedings.mlr.press/v80/wang18c.html.
12. Jeff Yan, Ahmad Salah El Ahmad. Captcha robustness: A security engineering perspective. IEEE Computer 44 (2), pp54-60, 2011.
13. Bin B. Zhu, Jeff Yan, Guanbo Bao, Maowei Yang, Ning Xu: Captcha as Graphical Passwords - A New Security Primitive Based on Hard AI Problems. IEEE Trans. Inf. Forensics Secur. 9(6): 891-904 (2014)