Research on the Security of Visual Reasoning CAPTCHA

Yipeng Gao[1], Haichang Gao[1*], Sainan Luo[1], Yang Zi[1], Shudong Zhang[1], Wenjie Mao[1], Ping Wang[1], Yulong Shen[1] and Jeff Yan[2]

[1]*School of Computer Science and Technology, Xidian University*
[2]*Department of Computer and Information Science, Linköping University*

Abstract

CAPTCHA is an effective mechanism for protecting computers from malicious bots. With the development of deep learning techniques, current mainstream text-based CAPTCHAs have been proven to be insecure. Therefore, a major effort has been directed toward developing image-based CAPTCHAs, and image-based visual reasoning is emerging as a new direction of such development. Recently, Tencent deployed the Visual Turing Test (VTT) CAPTCHA. This appears to have been the first application of a visual reasoning scheme. Subsequently, other CAPTCHA service providers (Geetest, NetEase, Dingxiang, etc.) have proposed their own visual reasoning schemes to defend against bots. It is, therefore, natural to ask a fundamental question: are visual reasoning CAPTCHAs as secure as their designers expect? This paper presents the first attempt to solve visual reasoning CAPTCHAs. We implemented a holistic attack and a modular attack, which achieved overall success rates of 67.3% and 88.0% on VTT CAPTCHA, respectively. The results show that visual reasoning CAPTCHAs are not as secure as anticipated; this latest effort to use novel, hard AI problems for CAPTCHAs has not yet succeeded. Based on the lessons we learned from our attacks, we also offer some guidelines for designing visual CAPTCHAs with better security.

1 Introduction

Completely Automated Public Turing test to Tell Computers and Humans Apart (CAPTCHA) is a defensive system for distinguishing computers from humans. Since L. Von Ahn [50] proposed this technology in 2004, CAPTCHAs have become an almost standard security mechanism for defending against malicious computer programs and bots. Each type of CAPTCHA scheme corresponds to a specific AI problem that is difficult for current computer programs to solve but is easily solvable by humans.

*Corresponding author: Haichang Gao (e-mail: hchgao@xidian.edu.cn)

Text-based CAPTCHAs have long been the most widely used scheme because of their simple structure and low cost. Such a CAPTCHA relies on a text recognition problem to distinguish humans from computers [51]. To resist the attack, text-based CAPTCHAs are often specifically designed with anti-segmentation features and anti-recognition features [6]. However, with advances in segmentation and character recognition technologies, most text-based CAPTCHAs have been solved [15], [5], [45], [32], [55], [14], [56], [13], [4], [57], [60], and designers need to find a new way to achieve security. Subsequently, image-based CAPTCHAs have been proposed. The image-based scheme is more diverse in content and background, and thus, it seems to be more secure than the text-based scheme. However, with the rapid development of computer vision techniques, it has been proven that solving CAPTCHAs based on image or object recognition is not a challenge for a machine [18], [59], [44], [29], [12].

In recent years, with the development and extensive application of deep learning, computers have been expected to have excellent logical reasoning skills to understand complex tasks similar to humans, which has led to the emergence of visual reasoning tasks based on computer vision and natural language processing. Subsequently, visual reasoning CAPTCHAs have also emerged as a new direction of development in the security field. Tencent, China's largest online instant messaging provider, proposed a visual reasoning scheme named the Visual Turing Test (VTT) [52], as shown in Figure 1. It uses the VTT CAPTCHA in Tencent Waterproof Wall [46], which serves hundreds of millions of people every day. This was the first application of a visual reasoning CAPTCHA, and it appears more secure than previous schemes. There are also three CAPTCHA service providers, Geetest, NetEase, and Dingxiang, who have now also proposed visual reasoning CAPTCHAs to defend against bots. It is therefore natural to ask a fundamental question: are the visual reasoning CAPTCHAs, in fact, as secure as their designers expect?

To comprehensively analyze the security of CAPTCHAs based on visual reasoning, this paper first proposes a holis-

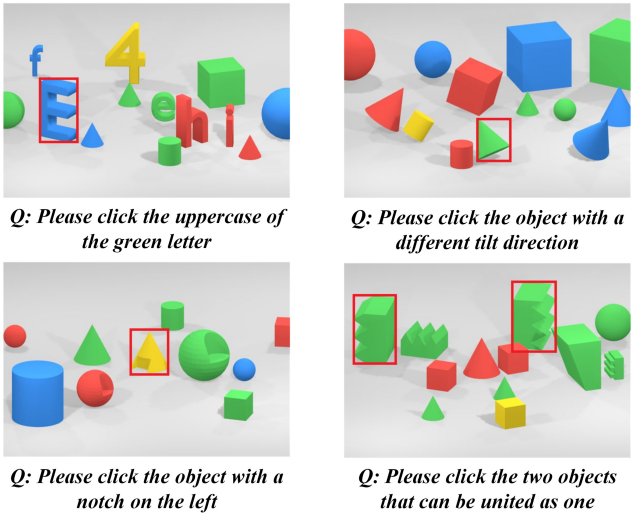


Figure 1: Samples of Tencent's VTT CAPTCHA.

tic method that consists of three modules: an input module extracts semantic features through a bidirectional long short-term memory (BiLSTM) network and visual features through a convolutional neural network (CNN); a reasoning module integrates the visual and semantic features to calculate the feature vectors of the possible answer objects; and an output module takes the output of the reasoning module as input to predict the final answer. Our holistic method is effective and robust. It achieves overall success rates of 67.3%, 66.7%, 77.8% and 86.5% on VTT, Geetest, NetEase, and Dingxiang CAPTCHAs, respectively. Through analysis, we found that most failures of our holistic method are related to abstract attributes that a computer program cannot obtain directly from an image, such as the literal meaning or pronunciations of characters.

Accordingly, to address the abstract attribute problem, we also propose a modular method. Its framework consists of four modules for query parsing, detection, classification, and integration. The query parsing module is responsible for transforming the text instruction of a VTT CAPTCHA into a series of reasoning steps, while the detection and classification modules predict the locations and visual attributes of all foreground objects. Finally, the integration module refers to the extracted reasoning steps to combine the visual and abstract attributes of objects to predict the final answer. The success rates of this modular method for VTT, Geetset, NetEase and Dingxiang CAPTCHAs are 88.0%, 90.8%, 86.2% and 98.6%, respectively.

Compared to the holistic method, the modular method is higher in accuracy but inferior in efficiency. Nevertheless, we have successfully broken visual reasoning CAPTCHAs. The high success rates of both of our attacks show that visual reasoning CAPTCHAs are not as secure as anticipated. Based on the lessons learned from our attacks, we summarize three guidelines for future CAPTCHA design. Our contributions are as follows:

- We present a comprehensive summary and analysis of the AI problems used as the basis of existing CAPTCHA schemes.
- We evaluate state-of-the-art visual reasoning CAPTCHAs and implement two successful attacks, which demonstrate that visual reasoning CAPTCHAs are not as secure as their designers hoped. To the best of our knowledge, this is the first attempt to solve visual reasoning CAPTCHAs in the industry.
- We summarize three guidelines (using a larger category set, making some occlusion, using more variations) and one promising direction for future CAPTCHA design.

2 AI Problems Underlying Existing CAPTCHA Schemes

The design principle of a CAPTCHA is to utilize the difference between the capabilities of human beings and machines in solving hard AI problems to defend against malicious bots or programs. The offensive and defensive nature of CAPTCHAs is thus manifested in a cycle of continuously cracking and designing new mechanisms addressing different AI problems. In this section, we mainly focus on the most widely used text-based and image-based CAPTCHAs and explore different hard AI problems hidden in different types of CAPTCHAs. Table 1 lists the different CAPTCHA schemes developed to date based on various AI problems, where the third column presents the defense strategies used and the last column shows typical examples.

2.1 Text-based CAPTCHAs

Early text-based CAPTCHAs adopted the character recognition task as the underlying hard AI problem and followed the anti-recognition principle for enhanced security. Gimpy and EZ-Gimpy are two such typical text-based CAPTCHAs. However, these two schemes have already been broken with high success rates [32]. Chellapilla et al. [7] further proved that computers are comparable to or even better than humans in recognizing distorted single characters. In fact, segmentation followed by recognition was the general process applied for early CAPTCHA cracking. Therefore, designers turned their attention to anti-segmentation algorithms, with the aim of preventing the successful extraction of characters from images. The most commonly used anti-segmentation schemes include crowding characters together (CCT), the hollow scheme, the two-layer, variable lengths, and background interference.

Unfortunately, all of these resistance mechanisms have also been broken. Gao's team [14] has proven that the hollow scheme can be broken using the color filling segmentation (CFS) algorithm. In 2017, they also proposed a method [13] of coping with the two-layer scheme. More recently, Tang et al. [45] proposed a pipeline method and broke a wide range of real-world CAPTCHAs with high success rates, thereby

Table 1: Different CAPTCHA schemes with different AI problems.

	AI Problems	Generation Methods	Representative Mechanisms
Text-based CAPTCHA	character recognition	distort, rotate, multi-font	Gimpy [32], EZ-gimpy [32]
	character segmentation	CCT, hollow, two-layer, variable length, etc.	Microsoft [13], Yahoo! [14]
Image-based CAPTCHA	object recognition	rich image categories	ASIRRA [11], Facebook [44]
	facial recognition	background embedding	ARTiFACIAL [40], FaceDCAPTCHA [22]
	image perception	orientation, size	What's up [21], DeepCAPTCHA [33]
	semantic comprehension	semantic relationship	SEMAGE [49], Google reCAPTCHA v2 [20]
	behavior detection	slider, notch	slider CAPTCHA [46], [16]
	adversarial perturbation	classification misleading	Adversarial CAPTCHA [37], [42]
	visual reasoning	logical relationship, attributes	VTT [52], Space CAPTCHA [16]

proving that the CCT scheme and background interference are also not secure. More innovatively, Zi et al. [60] proved that CAPTCHAs of this type can be completely broken under deep learning attacks without segmentation, indicating that anti-segmentation mechanisms, in general, are losing ground.

In addition to text-based CAPTCHAs designed with English letters and digits, Wang et al. [53] demonstrated that text CAPTCHAs based on large character sets, such as Chinese, Korean, and Japanese, are also not secure.

On the basis of the high success rates achieved to date, researchers have begun to emphasize efficiency in breaking CAPTCHAs. Other methods from the machine learning field have also been applied in cracking efforts, such as reduced training sets [17], the generative adversarial network (GAN)-based approach [57], and unsupervised learning and representation learning [47].

Overall, only limited space for improvement remains for text-based CAPTCHAs. Thus, CAPTCHA designers have gradually set their sights on the image domain.

2.2 Image-based CAPTCHAs

Image-based CAPTCHAs are the most popular alternative to text-based CAPTCHAs. Compared to the simple text-based scheme, image-based CAPTCHAs can contain more abundant information, with more categories and more diversity in image content. We simply categorize image-based CAPTCHAs based on different AI problems as follows:

CAPTCHA based on object recognition. Early image-based CAPTCHAs adopted object recognition as the underlying AI problem. This type of CAPTCHA usually asks users to identify specific images from several given categories. The robustness of an image-based CAPTCHA of this type depends on the number of object categories [59]. Evolving from ASIRRA [11] to the multiclassification CAPTCHAs of Google and Facebook, this principle has been widely adopted in subsequent image-based CAPTCHA design. However, each problem has been successfully solved [18], [44]. Currently, image CAPTCHAs based only on object recognition are not sufficient.

CAPTCHA based on facial recognition. The facial recognition task is also widely used as the underlying hard AI problem in image-based CAPTCHA design. ARTiFACIAL [40] requires users to click the corners of the eyes and mouth of a human face hidden in a complex background image. In FaceDCAPTCHA [22], a series of human faces are embedded in the background, and black color blocks are added to faces for enhanced security. However, both schemes have been successfully broken [29], [12]. The work of Uzun's team [48] also showed that current facial recognition services are insecure.

CAPTCHA based on image perception. The What's Up CAPTCHA proposed by Google [21], is based on identifying an image's upright orientation. Recently, Baidu and Dangdang [9] used a variant of What's Up CAPTCHA to defend against bots. It seems that image orientation perception remains a hard AI problem. The main limitation is that for a large number of images, orientation is difficult for both humans and computers. In addition, DeepCAPTCHA [33] distinguishes humans and bots based on depth perception. In this CAPTCHA, the user is required to arrange 3D objects in order of size (or depth) by clicking or touching them. The security of CAPTCHAs based on image perception is expected to be a subject of future work by both designers and attackers.

CAPTCHA based on semantic comprehension. Some CAPTCHAs [49], [20] capitalize on the human ability to com-

prehend image content and establish semantic relationships. These CAPTCHAs often ask users to select semantically related images from a given image set or select all areas that contain specified semantic information from the sections of a CAPTCHA image. The main limitation lies in the CAPTCHA generation stage. The definition of the correct relationships, the legal issues facing image collection, the time consumption required for image labeling, and the implementation of a regular updating strategy all pose large challenges.

CAPTCHA based on behavior detection. Slider CAPTCHA is a newly emerging type of CAPTCHA based on behavior detection. It asks the user to drag a slider to fill in a notch in a background image or simply to slide it from one side to another. For a machine, such a CAPTCHA essentially poses an object detection and behavior simulation problem. Zhao et al. [58] designed an algorithm based on the exclusive OR (XOR) operation to detect the notch position and mimic human behavior by leveraging common activation functions to bypass detection. They achieved success rates ranging from 96% to 100% on Geetest, Tencent, and NetEase slider CAPTCHAs. As an increasing number of protection mechanisms tend to detect abusive traffic based on user interactions with the website, not just the behavior when sliding the bar, the security of slider CAPTCHAs still needs further evaluation.

CAPTCHA with adversarial perturbation. It has been proven that deep neural networks are vulnerable to well-designed input samples, called adversarial examples [1], [19], which are imperceptible to humans but can easily fool deep neural networks. To further improve CAPTCHA security, Margarita [37] used adversarial examples for CAPTCHA generation within an object classification framework. In addition, adversarial examples were also adopted in the design process of reCAPTCHA v2 [20] to resist attacks based on deep learning. Shi et al. [42] proposed a framework for text-based and image-based adversarial CAPTCHA generation to improve the security of normal CAPTCHAs while maintaining similar usability. The combination of adversarial examples and CAPTCHAs is currently still in the exploration stage.

With the rapid development of the AI field, many other new types of CAPTCHA schemes have sprung up, such as reasoning puzzle CAPTCHA [34], word-order click CAPTCHA [36], scratch cards CAPTCHA [10], etc. Visual reasoning CAPTCHAs are also a new type of image-based CAPTCHA that relies on visual reasoning tasks, the combination of computer vision tasks and natural language processing tasks. The "visual reasoning" task includes multiple AI problems at the same time, such as object recognition, semantic comprehension, and relational reasoning. It shows a scene in which different objects have a logical relationship in position or content, and the answer needs to be obtained based on the common comprehension of text and images, which is more complicated than CAPTCHAs based only on object recognition or semantic comprehension. At present, research on visual reasoning CAPTCHAs is still lacking. We will discuss visual reasoning CAPTCHA and related research in detail in the next section.

3 Visual Reasoning CAPTCHAs

In this section, we first introduce existing visual reasoning schemes and their respective characteristics and then analyze existing methods to solve hidden AI problems behind the visual reasoning CAPTCHA. Finally, we illustrate the difference between the visual reasoning CAPTCHAs and the AI problem behind it and the difficulty of cracking.

3.1 Existing Schemes

Tencent first proposed a new CAPTCHA named VTT based on a visual reasoning task. Each VTT challenge consists of an image and a text instruction referring to the image. To pass the test, the user must understand the relationship expressed in the text instruction and click a specific region of the image. A VTT image usually contains 10 to 20 synthetic 3D objects. There are three possible types of challenges in VTT CAPTCHA:

An object's own attributes. The user must identify each object's visual attributes, including common attributes such as *geometric shape*, *color*, and *size*, as well as subtle attributes such as *tilt direction*, *fracture type*, *notch type*, and *character category*. Examples of related instructions include *"Please click the yellow cube," "Please click the object tilting to the left."*

A visual logical relationship. Related instructions may concern comparative relationships, e.g., *"Please click the biggest cylinder,"* or spatial relationships, e.g., *"Please click the cube left of the cone."*

An abstract logical relationship. Related instructions may invoke 1) synonym or antonym, e.g., *"Please click the two characters with opposite meanings"*; 2) pronunciation, e.g., *"Please click the Chinese characters with pronunciation 'bai'"*; 3) character components, e.g., *"Please click the Chinese characters with component '亻'"*; 4) uppercase or lowercase, e.g., *"Please click the uppercase of the green letter"*; 5) numerical sorting, e.g., *"Please click the numbers from the smallest to biggest"*. Such problems are more difficult for a machine to solve since the machine cannot obtain the necessary knowledge from either the image or the text instruction.

Geetest, a worldwide CAPTCHA service provider, has also designed a simplified scheme called Space CAPTCHA [16]. It looks almost the same as VTT but involves only regular geometries. The challenges contain only common attributes and spatial relations. Each image contains 7 to 10 objects. The prompts concern only the *colors*, *shapes*, *sizes*, and *spatial relationships* of regular geometric objects. However, the object categories and prompt formats are all different from those

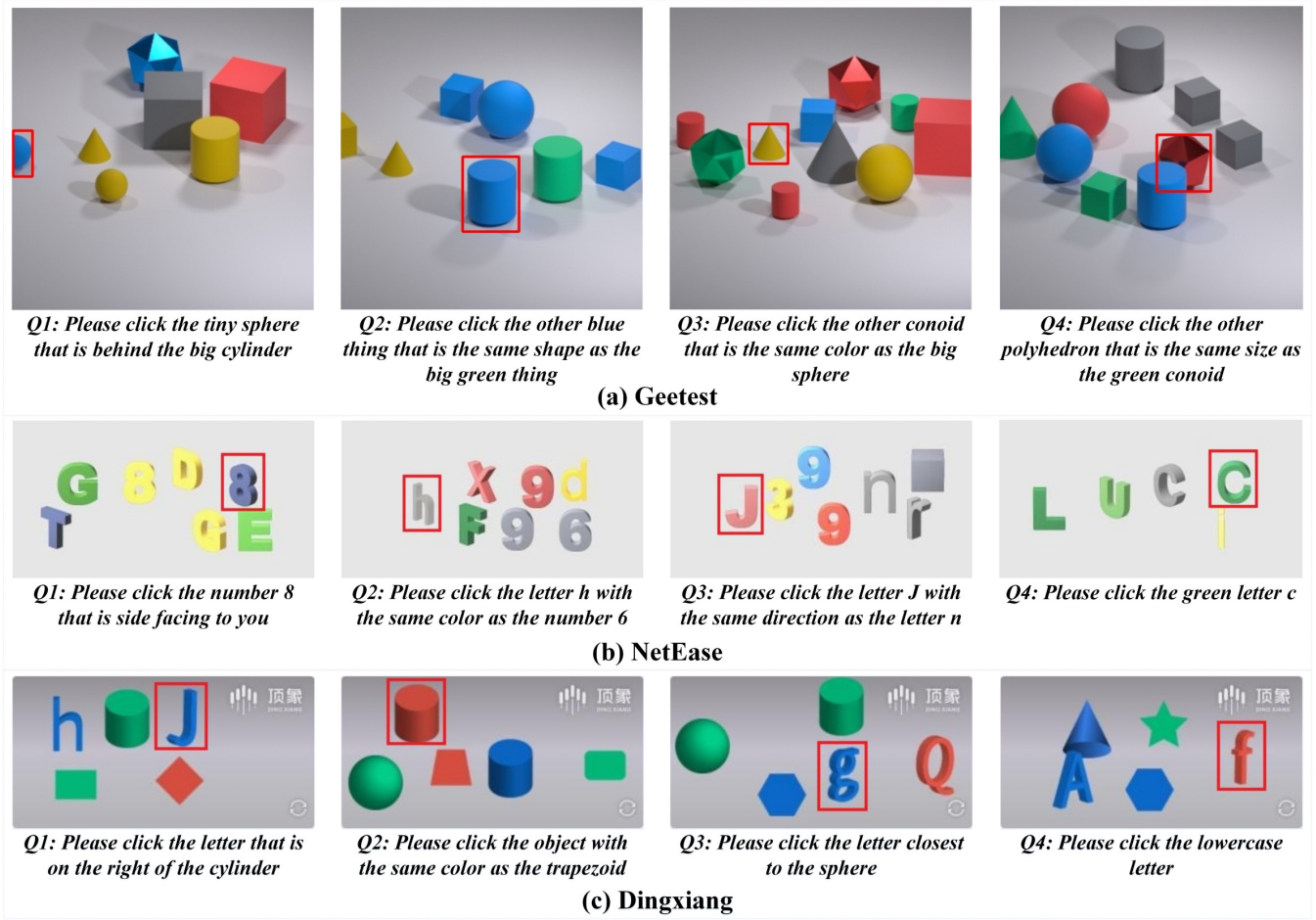


Figure 2: Samples of more visual reasoning CAPTCHAs.

of VTT. In addition, occlusion is more common in Space CAPTCHA. For example, the answer object in Figure 2(a.*Q1*) is incomplete. In Figure 2(a.*Q4*), the polyhedron is blocked by the blue cylinder. In addition, it is often the case that the relative spatial relationships are not very clear in challenges concerning location. For instance, it is difficult to distinguish whether the blue sphere in Figure 2(a.*Q1*) is behind the cylinder.

NetEase [35] and Dingxiang [10] have also designed spatial reasoning CAPTCHAs. Both contain fewer objects, attributes, and visual logical relationships and no abstract logical relationships (shown in Figure 2(b) and 2(c)).

NetEase's visual reasoning CAPTCHA contains regular geometric shapes, English letters, and digits. Each image usually contains 5 to 7 objects. The prompts mainly focus on objects that are "the same color", "side facing", and "with the same direction".

Dingxiang's CAPTCHA includes planar graphics, regular geometric shapes, and English letters. Each image shows 5 objects. The prompts concern only the locations (e.g., *up*, *down*, *left*, *right*, *closest to*) of objects or objects of the same color.

The main object categories in the existing visual reasoning schemes are shown in Table 2.

Table 2: Main object category in the existing visual reasoning schemes.

	VTT	Geetest	NetEase	Dingxiang
Regular geometries	✓	✓	✓	✓
Chinese characters	✓	-	-	-
English letters	✓	-	✓	✓
Digits	✓	-	✓	-

3.2 Related Work and Key Issues

Visual reasoning tasks have emerged as a basis for evaluating the logical reasoning abilities of AI systems. Three datasets, DAQUAR [31], VQA [3], and CLEVR [26], have been built as standard datasets for visual reasoning tasks that require a computer to infer an answer from an image for a given text-based prompt concerning spatial and semantic relationships. Simply put, the input problems for visual reasoning tasks are relatively difficult, involving multilevel relationships

among objects. Therefore, to solve such a task, an AI model needs reasoning capabilities, and a neural module network is an effective method. Methods of this kind make full use of the composability of language. Many small neural modules responsible for specific functions such as detection and location are defined, and the input problem is then parsed into a combination of modules composing a program that can be executed to obtain the answer to the prompt. [25], [8], [43] are several typical reasoning models.

However, the current AI solutions to visual reasoning problems are not sufficient for solving visual reasoning CAPTCHAs. The reason is that solving the CAPTCHA is not exactly equivalent to solving the underlying visual reasoning problem. Specifically, measures such as changing the form of the prompts and applying the click mechanism make the task of cracking this type of CAPTCHA different from that of simply solving a visual reasoning problem, as these measures may invalidate the reasoning mechanism. Therefore, how to deal with such changes is a difficult point to consider.

In addition, most of the current technologies for cracking CAPTCHAs are only aimed at solving specific mechanisms, and some general cracking methods tend to focus on the commonality of different CAPTCHAs. The novel AI problem involved in visual reasoning CAPTCHAs, i.e., the in-depth analysis and inference of the question to determine the answer, is the first time used in the CAPTCHA field. The simple convolutional network and long short-term memory network applied to previous text and image cracking methods have no way to understand some meanings more deeply. Thus, the inapplicability of past technologies to new mechanisms is also a bottleneck that we need to address. In fact, VTT designers have evaluated its security by implementing an attack experiment with a relation network and achieved only a 4.7% success rate [52].

Does this mean that the security of the visual reasoning CAPTCHAs is as their designers expected? In the following section, we present an in-depth analysis to answer this question.

4 Holistic Approach

In this section, we introduce a holistic attack on the representative visual reasoning CAPTCHA, VTT. After introducing this attack, we conduct a comprehensive analysis of its results and the reasons for its failure cases. We also attacked visual reasoning schemes designed by Geetest, NetEase, and Dingxiang to demonstrate the universal capabilities of our method. To evaluate the robustness of our attack, we also present two groups of experiments addressing higher logical complexity and new categories.

4.1 Model structure

The VTT CAPTCHA and the traditional visual reasoning task are two distinct tasks. The former is a reasoning detection task that requires the correct object to be located, while the latter requires giving a text answer. To solve the VTT CAPTCHA, we modify the MAC model [25], which achieved state-of-the-art performance on the CLEVR dataset in 2018, to output an object detection result rather than a text answer.

As long as the user clicks on any pixel of the target object in the VTT image, the system will determine the user to be a human. Inspired by YOLO-v3 [38], we evenly divide each image into a 14×14 grid and, for each grid cell, predict whether the center coordinates of the object of interest are located in that grid cell. Figure 3 depicts an outline of our holistic model, which consists of an input module, a reasoning module, and an output module.

1) Input module. The input module is designed to extract semantic features and global visual features. For the semantic feature extractor, we adopt the original BiLSTM [41] network to process the word embeddings of the text instruction. The output states of the BiLSTM network, $\boldsymbol{cw}_1, \boldsymbol{cw}_2, \ldots, \boldsymbol{cw}_s$, represent each word in the instruction string, whose length is $\boldsymbol{s}$. The final hidden states from the backward and forward directions of the BiLSTM network are concatenated to form the global semantic feature vector of the whole text instruction, denoted by $\boldsymbol{q}$. To extract the global visual feature vector $\boldsymbol{f}$, we replace ResNet-101 with ResNet-50 [23], which allows a larger batch size and provides a faster training speed and better prediction performance.

2) Reasoning module. The reasoning module is the core of our holistic model. It has a recurrent structure and consists of a sequence of elementary reasoning cells. Our reasoning cell follows the working principle of the MAC cell [25]. It contains two basic units: a control unit and a memory unit. The control unit receives both the semantic feature vector $\boldsymbol{q}$ and the control state $\boldsymbol{C}_{i-1}$ from the previous step to calculate the updated control state $\boldsymbol{C}_i$. It determines which part of the text instruction is the most relevant to each reasoning step. The memory unit is responsible for taking orders from the control unit and identifying the most important part $\boldsymbol{u}_i$ from the global visual feature vector $\boldsymbol{f}$. Then, the memory unit incorporates the previous memory state $\boldsymbol{M}_{i-1}$ and $\boldsymbol{u}_i$ to obtain the updated memory state $\boldsymbol{M}_i$. The memory state represents the most relevant visual information in each step.

Compared to the original MAC cell, our reasoning cell lacks a write unit. The write unit of the MAC cell is designed to integrate information retrieved from the global visual feature vector with the current memory state. The intermediate result of the write unit represents the current information of the reasoning process. For the CLEVR dataset, the model needs to output a text description of the answer. In contrast, VTT CAPTCHA requires the model to predict the coordinate information of the answer object. Due to this special require-

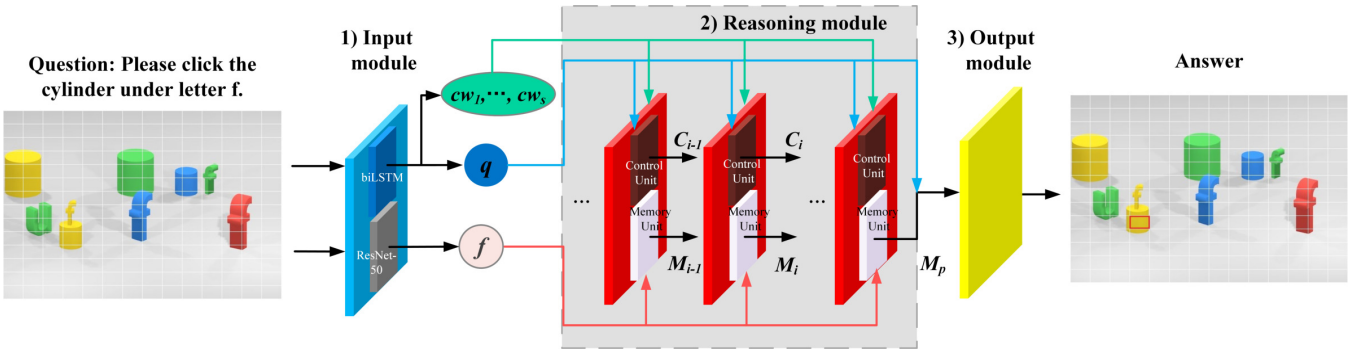


Figure 3: Framework of the holistic model. (The final answer is labeled with a red rectangle)

ment of the VTT CAPTCHA, using the memory state from the memory unit instead of the output of a write unit to predict the answer grid cell is a more reasonable approach.

3) Output module. The output module receives the global text representation $\boldsymbol{q}$ and the final memory state $\boldsymbol{M}_p$ as inputs. Then, $\boldsymbol{q}$ and $\boldsymbol{M}_p$ are concatenated together and passed through a classifier that consists of two fully-connected layers, one ReLU layer, and one softmax layer. The dimensions of the last fully-connected layer are modified to $196(14 \times 14)$ to allow the model to predict the probability distribution over all candidate grid cells. After normalization by the softmax layer, the grid cell with the highest score is the final prediction of our model.

4.2 Experiments and analysis

1) Implementation details

Data preparation. First, we collected 13,500 VTT CAPTCHA instruction-image pairs from the Internet [46]. The labeling task was to label the bounding box of the answer. In most cases, there was only one answer object for a given challenge. It took less than one day for five of this paper's authors to finish the labeling task. For each VTT test, the final feature map has dimensions of 14×14, so every test image was evenly divided into 14×14 grid cells to map each position in the feature map to the original image. Then, we wrote a simple Python program to calculate the grid cell containing the central pixel of the answer object. Accordingly, the calculated grid cell was labeled the ground truth for the VTT test. Finally, we divided the samples into a training dataset (10,000), a validation dataset (2,500), and a test dataset (1,000).

Training. Each image was normalized to 224×224 pixels before being processed by the model. The text instructions were embedded in a 300-dimensional space. The dimensionality of the hidden states (the control state and memory state) of our model was set to 512. We combined 16 reasoning cells to build the core reasoning module. A variable dropout strategy and exponential linear unit (ELU) activation functions were used throughout the network. In the training phase, the model was trained by minimizing the softmax cross-entropy loss with the Adam [28] strategy for 25 epochs on an NVIDIA GTX 1080 GPU.

Table 3: Proportions and success rates of different answer questions.

Answer object	Proportion	Success rate
Regular geometries	35.5%	78.5%
Chinese characters	30.2%	32.9%
English letters	18.2%	83.6%
Digits	16.1%	76.2%
Total	100.0%	67.3%

2) Experimental results

Our holistic approach achieved an average success rate of 67.3% on the test dataset. Moreover, the average processing time for each CAPTCHA was less than 0.05 seconds, which is 120 times faster than a human being [52].

Although the success rate of 67.3% is encouraging, it also indicated that our approach failed on some CAPTCHAs. Based on the categories of the answer objects, instances of the VTT CAPTCHA can be roughly divided into four classes: those based on regular geometric objects, Chinese characters, English letters, and digits. Table 3 lists the proportions and success rates for the different challenge types. From the proportions, we find that challenges concerning regular geometric objects make up the largest part of the entire dataset, followed by challenges concerning Chinese characters. Challenges addressing English letters and digits are fewer in number. In this experiment, the success rate for challenges based on English letters was the highest, at 83.6%. The success rates for challenges based on regular geometric objects and digits were 78.5% and 76.2%, respectively, while for challenges related to Chinese characters, only a 32.9% success rate was achieved because of the diversity of the character classes.

We comprehensively analyzed the reasons for the failure cases of our holistic method and found that the main reasons for failure are different for different challenge types. Some failure samples for our holistic model are shown in Figure 4. The failures of our holistic method can be attributed to four main causes:

Classification error. As shown in Table 4, classification

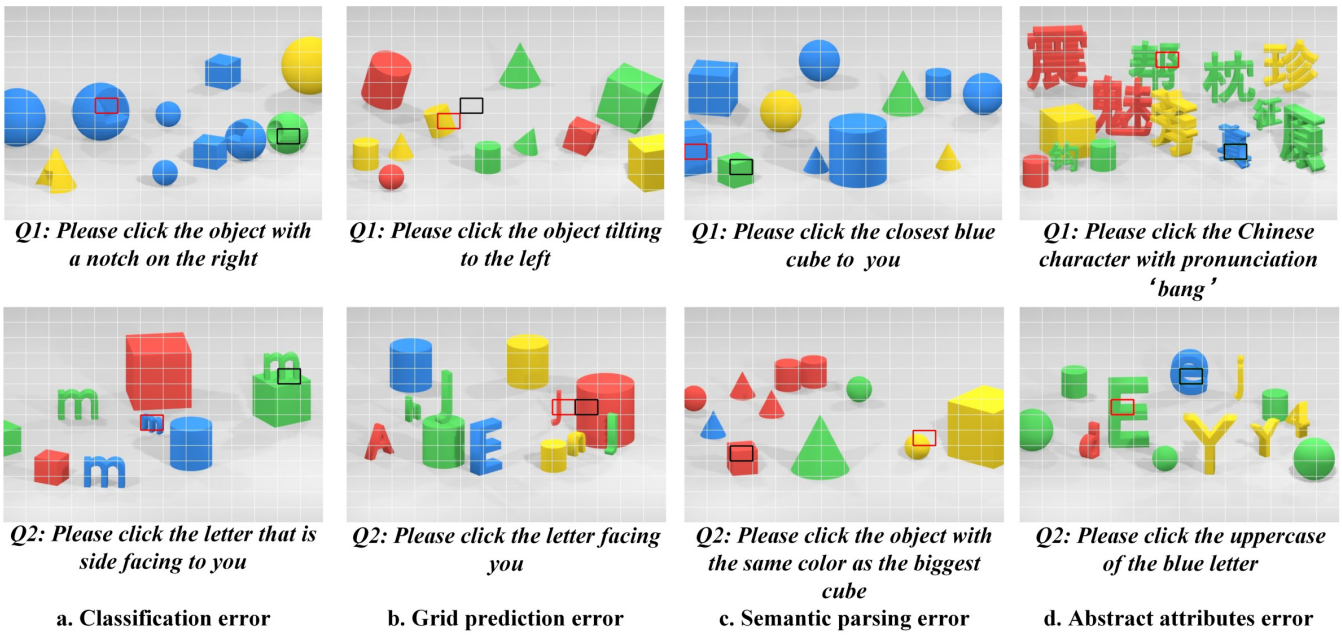


Figure 4: Failure samples for our holistic method.

Table 4: Error distribution(%) for the holistic method.

Answer object	CE	GPE	SPE	AAE	Others
Regular geometries	69.6	15.9	8.7	0	5.8
Chinese characters	18.1	0	0	81.9	0
English letters	20.2	17.0	11.4	45.7	5.7
Digits	15.5	26.2	20.0	38.3	0

* *Abbreviations in Table 4: CE (classification error), GPE (grid prediction error), SPE (semantic parsing error), AAE (abstract attribute error)*

errors account for 69.6% of attack failures on challenges concerning regular geometric objects. The subtle attributes of regular geometric objects include the tilt direction, notch type, and fracture type. For English letters and digits, classification errors are responsible for 20.2% and 15.5%, respectively, of all attack failures. The only subtle attribute of the relevant objects in these two categories is the side facing direction. For Chinese characters, classification errors account for 18.1% of attack failures. In this category, subtle visual attributes exist in relatively few training samples compared to color, shape, and other common attributes. Our model can learn the features corresponding to common attributes for almost all types of samples, while some subtle attributes appear only in relation to specific challenges. Therefore, the performance of our model in recognizing these subtle attributes is slightly inferior (see the failure cases shown in Figure 4(a.*Q1*) (a.*Q2*)).

Grid prediction error. The design principle of our holistic attack simplifies the complexity of the task and improves the attack efficiency. However, this design will sometimes lead to inaccurate prediction, with the model incorrectly outputting a grid cell that is close but not identical to the answer grid cell (shown in Figure 4(b.*Q1*) (b.*Q2*)). Such grid prediction errors are responsible for 15.9%, 17.0%, and 26.2% of the failure cases on regular geometric objects, English letters, and digits, respectively.

Semantic parsing error. Another failure cause is that our holistic model fails to extract the logical relationships expressed in the natural language instructions. Taking Figure 4(c.*Q1*) as an example, the model successfully recognized the "cube closest to the user" but missed the color information "blue" and instead found a "green" one, resulting in failure. Such semantic parsing errors are responsible for 8.7% of the failures on regular geometric objects, 11.4% of the failures on English letters, and 20.0% of the failures on digits.

Abstract attribute error. Table 4 shows that failure to identify abstract attributes is responsible for 81.9% of the failures on challenges based on Chinese characters. According to our manual count, most of the Chinese-based VTT CAPTCHA instances in our dataset involve abstract attributes. Because there are thousands of Chinese character classes, the numbers of classes of synonyms or antonyms, pronunciations, components, and other attributes are even larger. The mapping relationships between the characters and their abstract attributes are independent of the presented image and text instruction themselves. Therefore, it is not surprising that our model failed to establish the relevant mapping relationships between Chinese characters and their abstract attributes (as shown in Figure 4(d.*Q1*)). The high proportions of failures related to abstract attributes for English-based and digit-based CAPTCHAs can be attributed to similar reasons: some of these CAPTCHAs involve the mapping between lowercase and uppercase letters (as shown in Figure 4(d.*Q2*)), while some relate to the sorting of digits. For English-based and digit-based tests, abstract attribute errors account for 45.7%

and 38.3%, respectively, of all failure cases. By contrast, 0% of the failures on regular geometric objects are related to abstract attributes because these objects have only common attributes and subtle attributes.

Table 5: Attack results for different visual reasoning CAPTCHAs.

	VTT	Geetest	NetEase	Dingxiang
Success Rate	67.3%	66.7%	77.8%	86.5%

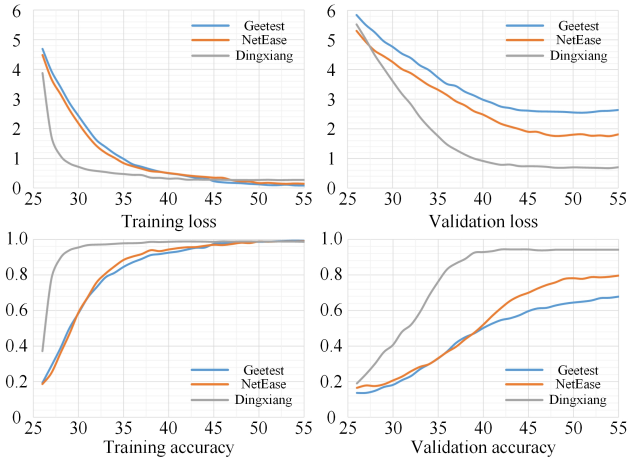


Figure 5: Loss and accuracy during the training and validation phases of Geetest, NetEase and Dingxiang.

4.3 More visual reasoning schemes

We also used the holistic method to attack the other three visual reasoning CAPTCHAs.

We collected 5,000 prompt-image pairs for each scheme from Geetest's website [16], NetEase's website [35], and Dingxiang's website [10]. A total of 4,000 samples were used for training, 500 were used for validation, and 500 were used for testing. The split of the dataset was randomly determined. We loaded the VTT baseline model and further trained it to fine-tune the holistic model for the new schemes. As shown in Table 5, the final attack results are 66.7%, 77.8%, and 86.5% successful, comparable to or better than the VTT attack results. For Geetest's Space CAPTCHA, although only regular geometric objects are involved, the attack success rate is lower than that of NetEase and Dingxiang. One of the reasons is that Geetest's Space CAPTCHA contains more objects in a challenge, and some of them are partially occluded by other objects. The other reason is that the combination of object attributes contained in the question is more abundant, which increases the difficulty of reasoning. In contrast, NetEase's and Dingxiang's CAPTCHAs contain richer categories, but the question is more straightforward, lower in complexity, and involves fewer types. The loss and accuracy on the Geetest, NetEase and Dingxiang samples during the training and validation phases are shown in Figure 5.

4.4 Robustness analysis

The experimental results discussed above show our holistic method's great ability to address the visual reasoning task in existing VTT CAPTCHAs. To test the robustness of our holistic model when faced with new variations, we conducted two groups of supplementary experiments.

1) Robustness to higher visual logical complexity

For the original VTT prompts, the user needs to refer to only one object to identify the answer object. For example, for the instruction *"Please click the blue cube that is on the right of the blue cone,"* the user needs to refer to the location of the blue cone to find the answer blue cube to its right. To test the robustness of our model to prompts with higher visual logical complexity, we extended the number of reference objects to 2 and 3. For instance, the instruction *"Please click the green cone that is on the right side of the green cone left of the red cube"* has two reference objects. It should be noted that we performed this robustness experiment after developing the modular attack. Considering that the logical reasoning task in the VTT CAPTCHA is similar to that on the CLEVR dataset, we modified the generation code of CLEVR [26] to generate this new type of VTT prompt in accordance with the image information we prepared for the modular attack.

We used 1,500 instruction-image samples (1,300 as the training dataset and 200 as the validation dataset) to fine-tune the baseline model for 2 and 3 reference objects and then evaluated the performance of the two fine-tuned models on their respective 500 test samples, which had the same distribution as the samples based on geometric objects in the baseline evaluation. The attack success rates of the two fine-tuned models were 45.0% and 42.3%. Compared to the 78.5% success rate of the baseline model, the fine-tuned results were slightly lower but still acceptable. The results show that despite the greatly increased logical complexity of the VTT instructions, with only a small number of newly labeled samples to train the baseline model, our holistic model still performs well in breaking the VTT CAPTCHA under the criterion of a 1% attack success rate [5].

2) Robustness to new object categories

Introducing new object categories into the VTT CAPTCHA design is a simple but valid way to defend against attacks from adversaries. In fact, each Chinese character class can be considered an individual category. Therefore, in this section, we used Chinese character classes to analyze the robustness to new object categories.

First, we removed all Chinese samples used in the base experiment and retrained our model in the same way as before. Without Chinese characters, the new model achieved 77.2%, 78.9%, and 85.7% success rates for challenges based on regular geometric shapes, English letters, and digits, re-

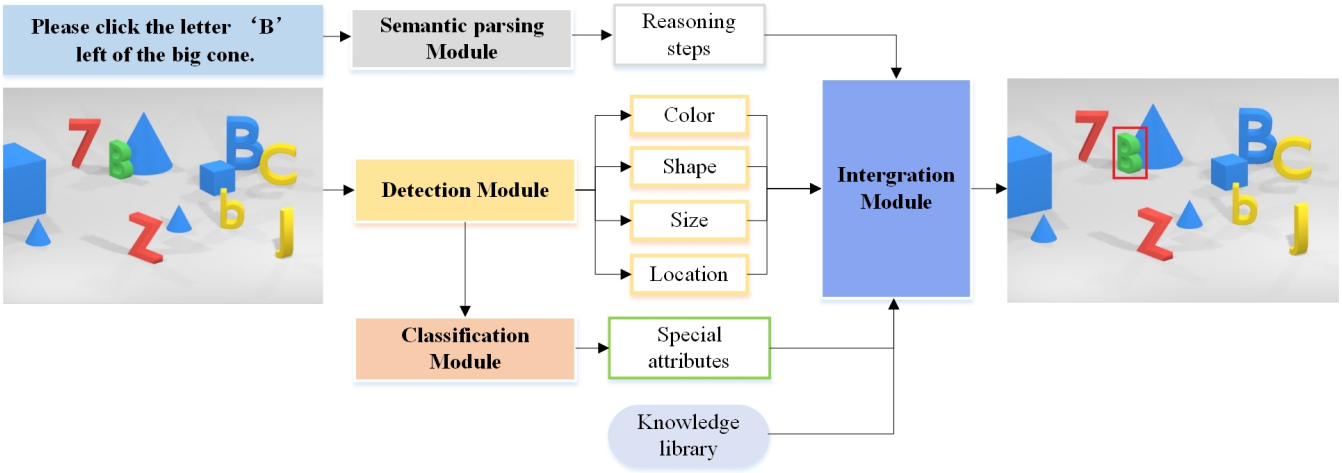


Figure 6: Framework of the modular approach.

spectively. The final success rate on the test dataset without Chinese samples was 77.9%. Then, we selected another 1,500 images (1,300 as the training dataset and 200 as the validation dataset) containing 100 Chinese character classes to generate corresponding visual reasoning based instructions for each image in a manner similar to the first robustness experiment. Note that the instructions were all based on common attributes rather than abstract attributes of Chinese characters.

After the new model was fine-tuned, the attack success rate on the 500 Chinese character challenges was 69.7%, showing the high robustness of our holistic attack to new object categories. This result is higher than the 32.9% success rate achieved in the base experiment. The reason is that the model needed to learn only the common attributes from 100 Chinese character classes represented in 1,500 images rather than many abstract attributes of thousands of Chinese character classes represented in nearly the same number of samples.

In summary, despite an increase in the visual logical complexity of the challenges or the introduction of new object categories, as long as the CAPTCHA is still based on the visual reasoning task, our method is able to achieve a high attack performance after fine-tuning on only a small number of newly collected CAPTCHA samples.

5 Modular Approach

Our holistic network has shown remarkable performance in breaking visual reasoning CAPTCHAs. However, when a CAPTCHA involves abstract attributes, such as *synonyms* or *antonyms*, *pronunciations*, or *components*, our holistic model does not work well. If we could manage to obtain the abstract attributes of all foreground objects and then integrate them into the process of completing the visual reasoning task, this problem could be solved. Based on this idea, we developed a modular method.

5.1 Model structure

The framework of our modular method is shown in Figure 6. It consists of four modules for semantic parsing, detection, classification, and integration. The semantic parsing module is responsible for inferring the reasoning steps necessary to complete the task. The detection and classification modules locate each foreground object and extract common attributes such as the *color*, *shape*, and *size*. The integration module then refers to the extracted reasoning procedure and aggregates all of the objects' attributes to predict the final answer.

1) Semantic parsing module

The semantic parsing module takes the raw text instruction $\boldsymbol{q}$ as its input and outputs the corresponding reasoning procedure $\boldsymbol{p}$. In essence, transforming $\boldsymbol{q}$ to $\boldsymbol{p}$ is a sequence-to-sequence task. As shown in Figure 7, the program generator network developed by Feifei's team [27] is adopted as the basis of our semantic parsing module. An encoder takes the raw text instruction $\boldsymbol{q}$ as its input and extracts its semantic features. A decoder then takes these semantic features to predict the corresponding program $\boldsymbol{p}$. Both the encoder and the decoder adopt a two-layer long short-term memory (LSTM) architecture as their core structures.

Step 1. The encoder first embeds the discrete words $< \boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_t >$ of the natural language instruction into 300-dimensional vectors $< \boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_t >$ through an embedding layer with weights $\boldsymbol{W}_x$:

$$\boldsymbol{x}_i = \boldsymbol{W}_x \cdot \boldsymbol{v}_i \tag{1}$$

All of these word vectors $< \boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_t >$ are then input into a two-layer LSTM with 256 hidden units in sequence. The reason for the choice of a two-layer structure instead of a single-layer structure is that it allows the network to extract higher-order features and enhances the representation capability of the semantic parsing module. For step $\boldsymbol{i}$ in each time, an LSTM cell takes the preceding hidden state $\boldsymbol{h}_{i-1}$ and the current word vector $\boldsymbol{x}_i$ as its input and outputs the updated

hidden state $\boldsymbol{h}_i$:

$$\boldsymbol{h}_i = LSTMStep(\boldsymbol{x}_i, \boldsymbol{h}_{i-1}) \tag{2}$$

The hidden state $\boldsymbol{h}_t$ of the second LSTM layer in the final time step t is used as the input to the decoder. For the same reason as for the encoder, a two-layer LSTM structure is adopted as the framework for the decoder. However, the network weights are not shared between the encoder and the decoder.

Step 2. For step $\boldsymbol{i}$ in each time step, the decoder network first concatenates its output $\boldsymbol{o}_{i-1}$ from the previous time step with the encoder's final hidden state $\boldsymbol{h}_t$ through a learned embedding layer. This operation allows the model to predict the current ***program p*** by referring to the previous prediction and the global semantic information:

$$\boldsymbol{u}_i = \boldsymbol{W}_u[\boldsymbol{o}_{i-1}, \boldsymbol{h}_t] \tag{3}$$

Step 3. $\boldsymbol{u}_i$ is used to compute the hidden state of the decoder cell, $\boldsymbol{o}_i$:

$$\boldsymbol{o}_i = LSTMStep(\boldsymbol{u}_i, \boldsymbol{o}_{i-1}) \tag{4}$$

Step 4. $\boldsymbol{o}_i$ is passed through a softmax layer to compute a probability distribution over all ***programs***:

$$\boldsymbol{s}_i = softmax(\boldsymbol{o}_i) \tag{5}$$

Step 5. The prediction with the highest probability is regarded as ***program p***:

$$\boldsymbol{p}_i = argmax(\boldsymbol{s}_i) \tag{6}$$

It should be noted that the semantic parsing module is responsible only for transforming the input text instruction into a sequence of ***programs***. The specific function of each ***program*** will be discussed in regard to the integration module.

2) Detection module

The task of the detection module is to locate the positions of all foreground objects. Faster R-CNN [39] is used as the detection module. Although there are other detection networks that perform better in terms of accuracy and efficiency, such as YOLO-v3 [38] and SSD [30], our detection task is relatively simple. Thus, the simple Faster R-CNN already satisfies our requirements.

In addition to locating the foreground objects, the detection network is able to perform some simple classification at the same time. Some common visual attributes, such as *colors*, *sizes*, and *shapes of regular geometries*, are also predicted by the detection module. After detection, the detected objects are cropped from the original images and sent to the classification module for further classification of subtle attributes.

3) Classification module

The function of the classification module is to recognize subtle visual attributes such as *notches*, *fractures*, *tilt directions* and *character categories*. SENet [24] is used as the classification module. By calculating the interdependencies among channels, this structure enables adaptive recalibration of the channelwise feature responses, thus greatly enhancing the representation power of the model and increasing the classification accuracy.

4) Integration module

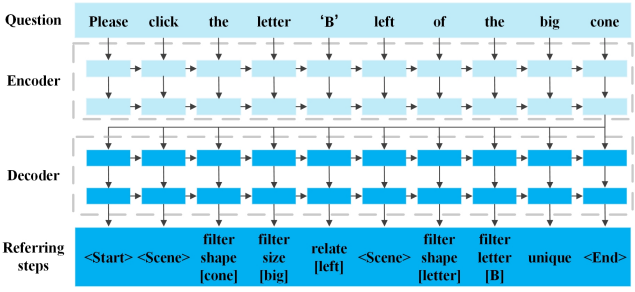


Figure 7: Structure of the semantic parsing module.

The three modules described above predict the reasoning procedure and visual attributes needed to solve CAPTCHAs. However, they cannot address abstract attributes invoked in the presented instructions. If we can establish the relevant mapping relationships between objects and their abstract attributes, the corresponding CAPTCHAs will be cracked. For each Chinese character object, we input its predicted character class into the online Xinhua Dictionary [54] to search for its *pronunciation*, *antonym*, and *component* attributes. The mappings between the uppercase and lowercase versions of English letters and the numerical sorting of numbers were established programmatically.

The extracted reasoning procedure for a CAPTCHA instance consists of a series of ***programs***, each of which represents a reasoning step. A ***program*** is responsible for filtering out redundant foreground objects. Different ***programs*** serve unique functions. After the processing of the ***program***, only objects with the required attributes remain. For example, the ***program filter_shape[cone]*** selects objects with the shape "cone" from among the objects remaining after the preceding ***program***. After a sequence of program-based filtration operations, the final remaining objects are the predicted answers.

Taking the CAPTCHA shown in Figure 8, with the instruction *"Please click the letter 'B' left of the big cone"* as an example, we describe the integration process in detail below. It consists of five ***programs*** in total. To clearly illustrate the integration process, the candidate answer objects are displayed in colors, while the eliminated objects are displayed in gray. The whole reasoning procedure is as follows:

a. Initially, all foreground objects are treated as candidate answers.

b. The first is ***program filter_shape[cone]***. Its function is to select all the objects with the shape "cone" from among all the candidate objects. As shown in Figure 8, only the cones are selected to be used as candidate answers to the next ***program***.

c. The second ***program***, ***filter_size[big]***, is responsible for selecting all objects with the size "big" from among the candidate objects output by the previous step.

d. The ***program relate[left]*** is slightly different. Instead of selecting candidate answers from the output of the last step, it treats the output of the last ***program*** as a reference to search for candidates among all the foreground objects. The output

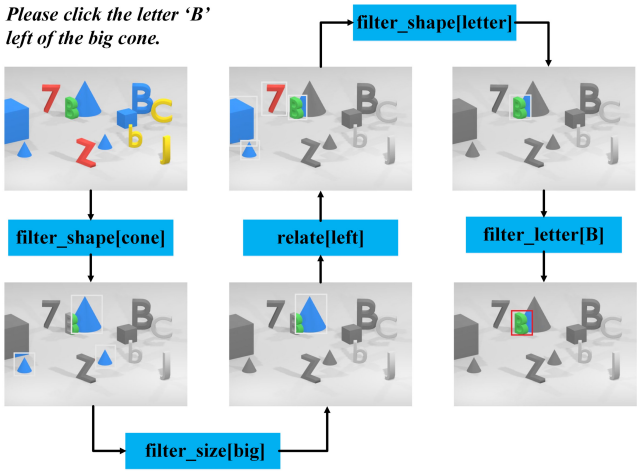


Figure 8: Integration process.

of the second ***program*** consists of a "big cone." Thus, the function ***relate[left]*** finds objects to the left of such objects.

e. After the ***program filter_shape[letter]***, only English letters remain.

f. The ***program filter_letter['B']*** searches among its candidate objects for objects equivalent to the letter "B."

g. Finally, after all of the ***programs*** have performed their filtration tasks, only the green letter "B" remains, which is the final answer of our model.

5.2 Experiment details

Data preparation. 1) Visual feature selection. We manually analyzed 2,000 VTT instruction-image pairs in our dataset and counted the visual attributes involved, including *color*, *shape*, *size*, *direction of rotation*, *notch type*, and *fracture type*. The number of classes of each attribute above is listed in Table 6. For the *tilt direction* attribute, "T1" and "T2" represent two different values. The naming principle for the values of the *notch type* attribute is similar. For the *fracture type* attribute, "Fi" and "F(-i)" can be joined together. **2) Instruction-image pairs preparation.** To reduce the labeling burden, we chose only 5,000 VTT images from among the training samples collected for the holistic experiment and labeled every foreground object in these images. Twenty members of our laboratory spent one day labeling all of the object attributes online. We needed only to select the corresponding attributes from option boxes instead of providing keyboard input. Each test image could be reused to generate multiple instructions. For this purpose, the generation code of CLEVR [26] was modified to automatically generate instructions in accordance with the labeled information and the preset VTT instruction templates. Instruction labeling was also automatically completed by means of the instruction generation code. Finally, 5,000 labeled images, each corresponding to 2 instructions (10,000 instructions in total), were prepared. It should be noted that the 5,000 selected images were not all randomly chosen. Instead, different types of images were selected in accordance with the category proportions in the holistic experiment, as shown in Table 3. Specifically, 1750 (35%), 1,500 (30%), 1,000 (20%), and 750 (15%) images were chosen for which the answer objects were regular geometric shapes, Chinese characters, English letters and digits, respectively. The test samples in the holistic experiment were reused in the modular attack test.

Table 6: Number of classes of different visual attributes.

Attribute	Number of Classes	Sample of Label
Color	4	Yellow, Red, Blue, White
Shape	924	Cube, r, 3, 田, ⋯
Size	3	Big, Medium, Small
Tilt direction	2	T1, T2
Notch	4	N1, N2, N3, N4
Fracture	8	F1, F2, F3, F4, F(-1), F(-2), F(-3), F(-4)

Training the semantic parsing module. We used 10,000 instruction and reasoning procedure pairs, denoted by ($\boldsymbol{q}$, $\boldsymbol{P}$), to train the semantic parsing module (8,500 as the training dataset and 1,500 as the validation dataset). For each instruction, the corresponding reasoning procedure was manually labeled. We used the cross-entropy loss to measure the difference between the model prediction $\boldsymbol{P}'$ and the true label $\boldsymbol{P}$ for instruction $\boldsymbol{q}$. During the training process, the Adam [28] strategy was used to optimize the model. The learning rate was set to 5×10^{-4}. The model was trained with a batch size of 64 for 16,000 iterations on an NVIDIA TITAN X GPU.

Training the detection module. A total of 5,000 images were used to train the detection module (4,500 as a training dataset and 500 as a validation dataset). Note that the detection module is responsible only for predicting object locations and simple visual attributes. The detection module was trained with a batch size of 8 and a learning rate of 5×10^{-3} for 32,000 iterations. The training hardware was the same as that for the semantic parsing module.

Training the classification module. According to the bounding boxes predicted by the detection module, we cut out all foreground objects from the original images and saved them as individual images. Each kind of subtle visual attribute was equally treated as one individual class regardless of the other attributes. The sizes of the training and validation datasets were 54,212 and 16,347, respectively. Each image was normalized to 224×224 pixels before being input to the model. The classification module was optimized using the stochastic gradient descent (SGD) strategy with a momentum of 0.9 and a batch size of 8. The learning rate was initially set to 1×10^{-4} and was decreased by a factor of 10,000 in every epoch. The model was trained for 10 epochs.

Table 7: Results of our modular attack.

Answer object	SPM	DM	CM	ASR
Regular geometries	100%	93.0%	90.0%	99.0%
Chinese characters	100%	96.6%	82.7%	80.0%
English letters	100%	98.5%	93.8%	83.7%
Digits	100%	99.0%	96.3%	94.7%
Overall accuracy	100%	95.0%	88.8%	88.0%

* *Abbreviations in Table 7: SPM (semantic parsing module), DM (detection module), CM (classification module), ASR (attack success rate)*

5.3 Evaluation

We ran our attack on 1,000 CAPTCHA challenges and achieved a success rate of 88.0% with an average speed of 0.96 seconds per challenge. To systematically analyze our method, we counted the failure cases of our attack (as shown in Table 7) and analyzed the causes.

Final accuracy. The accuracy for the challenges based on Chinese characters is the lowest due to their diversity and complexity. We observed an interesting phenomenon: although the detection accuracy and classification accuracy for geometric objects are not the highest, their overall accuracy is the best. One reason is that geometric objects do not have abstract attributes. Another is that during the process of cracking a visual-based CAPTCHA, the model does not need to recognize all foreground objects correctly; as long as the target object is recognized correctly, the challenge is considered cracked.

Semantic parsing module. The evaluation criterion for the program generator is that the prediction for a text instruction is considered correct only if every step of the predicted reasoning procedure is equal to the ground truth. Under this standard, the program generator achieved 100% accuracy. The program generator network has previously shown great power on the CLEVR task [27]. Thus, considering that the text instructions of the VTT CAPTCHA scheme involve fewer categories and much simpler logical relationships, this high accuracy is not surprising.

Detection module. The overall true positive rate (TPR) of detection of the Faster R-CNN module across all classes is 95.0%. We found that occlusion was the main cause of failure. Figure 9 shows a failure case of our detection module. The red bounding boxes represent the predictions of our model, and the green bounding box represents an object that was not correctly predicted. The blue cylinder in the green bounding box was not detected because its edge was partially blocked by a Chinese character.

Classification module. The overall accuracy of the classifier is 88.8%. As expected, the accuracy of Chinese characters is the lowest. The number of categories of Chinese characters is the largest, and tilt and occlusion effects make the classification problem even more challenging. Consequently, the classifier can easily misclassify these characters. Moreover,

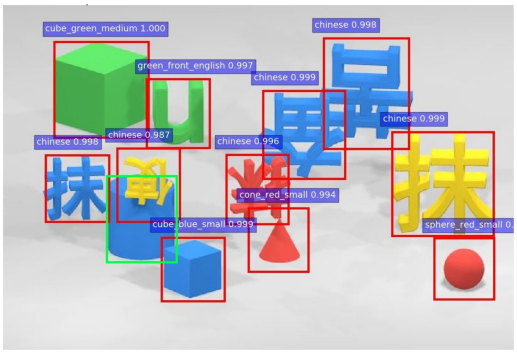


Figure 9: A failure case of the detection module.

Table 8: Results for different visual reasoning CAPTCHAs of the modular method.

	SPM	DM	ASR
Geetest	100%	95.7%	90.8%
NetEase	100%	93.5%	86.2%
Dingxiang	100%	95.2%	98.6%

* *Abbreviations in Table 8: SPM (semantic parsing module), DM (detection module), ASR (attack success rate)*

the classification accuracy for geometric objects is the second lowest. For geometric objects, the task of the SENet module is to classify their subtle attributes, such as *tilt direction*, *notch type*, and *fracture type*. These attributes are essentially local features relative to the shape of the object. For example, two distinct geometric objects might have the same notch type. As a result, the classifier must strip these local features from the various geometric shapes.

5.4 More visual reasoning schemes

1) Attack

We also used the modular method to attack the other three visual reasoning CAPTCHAs. The three schemes have much fewer categories than VTT CAPTCHA. To simplify our experiments, we removed the classification module and used the detection module to complete detection and classification tasks simultaneously. We used the data collected in Section 4.3 and annotated the data in the same manner as in Section 5.2. For each scheme, there are 4,000 samples for training the models of semantic parsing and detection modules, 500 samples for validation and 500 samples for testing. Table 8 list the experiment results. The final attack results are 90.8%, 86.2% and 98.6% for the Geetest, NetEase and Dingxiang schemes, respectively. This suggested the wide applicability of our method.

2) Usability Analysis

To visually express the quality of the proposed attack methods, we compared the attack results with actual humans from two aspects. On the one hand, considering that the CAPTCHA is used to distinguish humans from bots, we expect to quantitatively measure how close our attacks are to human per-

formance. On the other hand, we want to learn whether the problems difficult for machines to solve also apply to humans.

We applied a framework similar to that used in [52] to quantitatively evaluate the usability of the four tested CAPTCHAs. More specifically, we analyzed the usability of these schemes from the perspectives of success rate and response time. For each CAPTCHA mechanism, 2,500 samples containing prompts of various types in even proportions were selected for online deployment. All of these CAPTCHA prompt-image pairs were derived from the training and test datasets used for the security analysis.

In the usability experiment, we invited 50 participants whose ages ranged from 19 to 45 on our campus to take our online tests. We recruited volunteers online on the campus social network. All volunteers were composed of students and teachers from various majors, who have enough ability to solve such CAPTCHA schemes. To avoid the inherent biases, we ensure that these volunteers have not done similar CAPTCHA tests before. Everyone was required to complete the test independently. Each volunteer was asked to complete at least 40 CAPTCHA tests for each scheme. We received 2475, 1969, 2061, and 2361 valid records for the four CAPTCHA mechanisms of VTT, Geetest, NetEase, and Dingxiang, respectively. Table 9 lists the success rates and average response times for the different CAPTCHA schemes.

The response times for all four schemes are relatively short, with the longest being 10.7 seconds for Geetest CAPTCHA. The consensus is that a CAPTCHA should be completable by a human in no more than 30 seconds [40], and these CAPTCHAs satisfy this principle well. Both the short response times and the high pass rates prove that these CAPTCHAs all have good usability and that complex problems for machines do not have a significant impact on humans.

Our methods approach or even exceed the human pass rates, which proves the effectiveness of the attack. Following the criterion that a scheme is considered broken when the attacker is able to reach a precision of at least 1% [5], our method achieved a good attack effect.

Table 9: Usability analysis of different CAPTCHA schemes.

	VTT	Geetest	NetEase	Dingxiang
Response Time (s)	9.1	10.7	4.5	5.7
Std Dev of Response Time (s)	5.5	5.9	3.0	4.3
Human Pass Rate(%)	87.48	90.76	95.20	95.43

5.5 Ablation study

Our modular attack is based on a modular design principle. To fairly evaluate the contributions of each of the three modules of our attack, we performed an ablation study, as reported in this section.

Contribution of the semantic parsing module. In this test, we removed the semantic parsing module and used only the detection module to predict the locations of foreground objects. Then, we randomly selected one foreground object as the final answer. We implemented this attack strategy on the same 1,000 samples used to test our modular method, and the final success rate was 6.9%. The dramatic reduction in the success rate demonstrates the great significance of our semantic parsing module in the entire modular attack.

Contribution of the detection module. The basic requirement to solve a VTT CAPTCHA instance is to identify an area of the image as the answer. Without the detection module, an adversary must take a brute force strategy to attack the VTT CAPTCHA. Using this method, the final success rate was only 3.2%, showing that the detection module is indispensable for our modular attack.

Contribution of the classification module. In this test, we removed the classification module and trained a Faster R-CNN model to predict both the bounding boxes and the classes of all visual attributes (including subtle attributes) of the foreground objects. That is, for all objects, only the detection module was used to perform both the detection and classification tasks. In this way, our simplified modular method achieved a success rate of 45.9%.

As shown in Table 10, we further calculated the accuracy of the simplified modular method for each challenge category. The second column presents the final detection-classification results, and the last column shows the final success rate when the classification module is removed. In contrast to the results for Chinese characters, the final success rate for challenges based on geometric objects is still very high. The root cause lies in the fact that for Chinese characters, there are more object categories represented by the same number of training samples. Consequently, there are fewer training samples for each character class. Moreover, it is quite difficult for an object detection network to classify a large number of categories, especially categories that contain subtle properties. Therefore, it is not unexpected that the success rate for Chinese characters is the worst. Thus, the classification module is required. When our classification module is presented with the same number of samples for Chinese characters as for geometric shapes, it can achieve much better accuracy on Chinese character objects.

In summary, our classification module not only increases the overall success rate from 45.9% to 88.0% but, more importantly, can greatly increase the recognition accuracy when the number of training samples is limited.

6 Guidelines and Future Direction

Our experimental attacks on visual reasoning CAPTCHAs not only reveal their weaknesses and vulnerabilities but, more importantly, help us better understand what kinds of mechanisms or design features contribute to good security. Based

Table 10: Results of the ablation study.

Target object	Detection-classification rate	Attack success rate
Regular geometries	93.2%	89.9%
Chinese characters	24.2%	20.0%
English letters	89.7%	54.5%
Digits	91.6%	78.9%
Overall accuracy	77.3%	45.9%

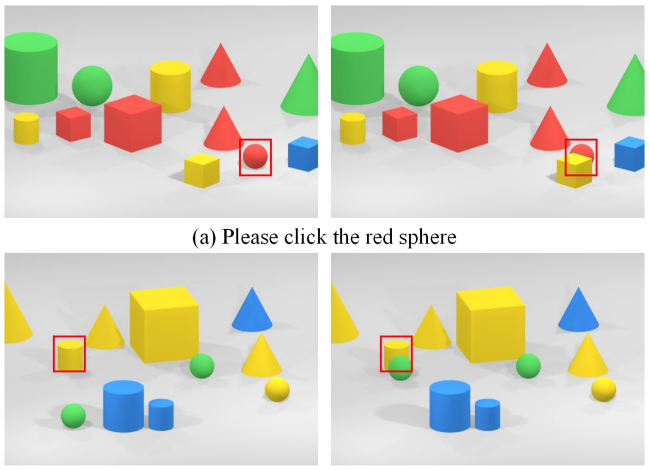
on the observation of the effectiveness of the different design features of visual reasoning CAPTCHAs, we summarize three guidelines for future CAPTCHA design that could make these types of CAPTCHAs harder to crack. We also evaluate the recommendations experimentally and continue to use commonsense knowledge in CAPTCHAs in future work.

Using a larger category set. As discussed above, using more categories in CAPTCHA design results in a larger theoretical solution space that a malicious bot must search and thus provides better security. To evaluate this guideline, we expanded the robustness experiments in Section 4.4 in the same experimental settings. Under the same amount of data, attacking VTT challenges containing 100 Chinese character classes is more difficult than attacking 50 Chinese character classes. The attack results in Table 11 strongly demonstrate our opinion. Meanwhile, according to our experimental results in Table 7, the classification accuracy for Chinese characters is the lowest among regular geometries, English letters, and digits, which indicates that using more classes indeed provides better defense against adversaries. Research by Algwil et al. [2] also corroborates our view. They have shown that in the context of recognition tasks, it is more demanding to attack CAPTCHAs with a Chinese mechanism than Roman character-based CAPTCHAs. One important reason is that the Chinese character set is a larger category set than English letters.

Table 11: The attack success rates of adding more categories.

	50 classes	100 classes
Attack Success Rate	77.7%	69.7%

Making some occlusion. Occlusion refers to the case in which the view of an object is partially blocked by another object. Making some occlusion will enhance the security of CAPTCHAs. To confirm this guideline, we set comparative experiments for no occlusion and occlusion of the answer objects, as shown in Figure 10. Meanwhile, we explore whether occlusion will affect human pass rates. We use one single question type and only regular geometries contained in images to simplify the experiments. Table 12 shows that the occlusion of the answer objects has significant impact on the machine attack results but has little impact on humans' ability



(a) Please click the red sphere

(b) Please click the yellow cylinder

Figure 10: Examples of no occlusion (left) and occlusion (right) of the answer objects.

to solve the CAPTCHA. The root cause lies in the fact that once part of an object is blocked, its edge information and part of its texture information are lost, in turn, which will affect the final prediction of the CNN model. In contrast, humans can infer the shape contour of an object by observing only a small fraction of it. Therefore, for visual perception-based CAPTCHAs, designers can make use of this defect of machine learning to enhance the security of CAPTCHAs.

Table 12: The attack success rate and human pass rate under different occlusion settings.

	No Occlusion	Occlusion
Attack Success Rate	86.0%	69.5%
Human Pass Rate	93.9%	92.9%

Using more variations. Variation refers to objects in the same category that appear subtly different but remain the same in their main outline and basic features. The experimental results of our holistic attack in Table 4 demonstrate that among all our attack failure cases, the recognition error rate is the highest for regular geometric objects. The root cause lies in the fact that more variations are introduced in the design of the geometric objects used in the VTT CAPTCHA, such as the *notch* and *slant* attributes. On the one hand, these attributes raise the difficulty for a model in recognizing the object category; on the other hand, recognizing these attributes themselves is even more challenging for a model than the category classification task. In fact, Zi et al. [60] argued that using a number of character fonts can greatly increase CAPTCHA security because it introduces more variations and requires a more robust attack model. Therefore, more variations can be introduced to enhance security.

Commonsense knowledge. Abstract concepts can be regarded as a type of commonsense knowledge. The inability of our holistic model to address abstract concepts resulted in

81.9%, 45.7% and 38.3% of its failures on VTT tests based on Chinese characters, English letters, and digits, respectively, as shown in Table 4, and our modular method can solve only a limited subset of challenges based on abstract concepts. However, the body of commonsense knowledge held by humans is nearly infinite. All these experimental results show that solving problems based on commonsense knowledge is indeed a complex task for current machine learning and deep learning algorithms. The high abstractness and infinite scope of commonsense knowledge greatly increase the problem complexity for a machine. We believe CAPTCHAs invoking commonsense knowledge will be a promising research direction.

7 Conclusion

In this paper, we explored the hard AI problems underlying current existing CAPTCHAs and found that conventional CAPTCHA schemes have been proven to be insecure. We comprehensively studied the security of one representative visual reasoning scheme, Tencent's VTT CAPTCHA, by means of a holistic attack and a modular attack and achieved success rates of 67.3% and 88.0%, respectively. To test the robustness of our method, we also conducted supplementary experiments on three other visual reasoning schemes. Our high success rates prove that the latest effort to use novel, hard AI problems (visual reasoning) for CAPTCHAs has not yet succeeded. We further summarized three guidelines for future vision-related CAPTCHA design and believe that in particular, the adoption of commonsense knowledge in CAPTCHA design has promising prospects.

Acknowledge

We would like to thank our shepherd David Freeman and the anonymous reviewers for their valuable suggestions for improving this paper. This paper was supported by the Natural Science Foundation of China under Grant 61972306 and sponsored by Zhejiang Lab (No. 2021KD0AB03).

References

[1] Naveed Akhtar and Ajmal Mian. Threat of adversarial attacks on deep learning in computer vision: A survey. *IEEE Access*, 6:14410–14430, 2018.

[2] Abdalnaser Algwil, Dan Ciresan, Beibei Liu, and Jeff Yan. A security analysis of automated Chinese turing tests. In *Proceedings of the 32nd Annual Conference on Computer Security Applications*, pages 520–532, 2016.

[3] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. VQA: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433, 2015.

[4] Elie Bursztein, Jonathan Aigrain, Angelika Moscicki, and John C Mitchell. The end is nigh: Generic solving of text-based CAPTCHAs. In *8th {USENIX} Workshop on Offensive Technologies ({WOOT} 14)*, 2014.

[5] Elie Bursztein, Matthieu Martin, and John Mitchell. Text-based CAPTCHA strengths and weaknesses. In *Proceedings of the 18th ACM conference on Computer and communications security*, pages 125–138, 2011.

[6] Elie Bursztein, Angelique Moscicki, Celine Fabry, Steven Bethard, John C Mitchell, and Dan Jurafsky. Easy does it: more usable CAPTCHAs. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 2637–2646, 2014.

[7] Kumar Chellapilla, Kevin Larson, Patrice Y Simard, and Mary Czerwinski. Computers beat Humans at Single Character Recognition in Reading based Human Interaction Proofs (HIPs). In *CEAS*, 2005.

[8] Xinlei Chen, Li-Jia Li, Li Fei-Fei, and Abhinav Gupta. Iterative visual reasoning beyond convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7239–7248, 2018.

[9] Dangdang. Rotation CAPTCHA in Dangdang login website. https://login.dangdang.com/signin.aspx?returnurl=http%3A//www.dangdang.com/.

[10] Dingxiang. Dingxiang's CAPTCHAs website. https://www.dingxiang-inc.com/business/captcha.

[11] Jeremy Elson, John R Douceur, Jon Howell, and Jared Saul. ASIRRA: a CAPTCHA that exploits interest-aligned manual image categorization. In *ACM Conference on Computer and Communications Security*, volume 7, pages 366–374, 2007.

[12] Haichang Gao, Lei Lei, Xin Zhou, Jiawei Li, and Xiyang Liu. The robustness of face-based CAPTCHAs. In *2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing*, pages 2248–2255. IEEE, 2015.

[13] Haichang Gao, Mengyun Tang, Yi Liu, Ping Zhang, and Xiyang Liu. Research on the security of microsoft's two-layer CAPTCHA. *IEEE Transactions on Information Forensics and Security*, 12(7):1671–1685, 2017.

[14] Haichang Gao, Wei Wang, Jiao Qi, Xuqin Wang, Xiyang Liu, and Jeff Yan. The robustness of hollow CAPTCHAs. In *Proceedings of the 2013 ACM SIGSAC conference*

on Computer & communications security, pages 1075–1086, 2013.

[15] Haichang Gao, Jeff Yan, Fang Cao, Zhengya Zhang, Lei Lei, Mengyun Tang, Ping Zhang, Xin Zhou, Xuqin Wang, and Jiawei Li. A Simple Generic Attack on Text CAPTCHAs. In *23rd Annual Network and Distributed System Security Symposium, NDSS 2016, San Diego, California, USA, February 21-24, 2016*, 2016.

[16] Geetest. The Geetest website. https://www.geetest.com/en/demo.

[17] Dileep George, Wolfgang Lehrach, Ken Kansky, Miguel Lázaro-Gredilla, Christopher Laan, Bhaskara Marthi, Xinghua Lou, Zhaoshi Meng, Yi Liu, Huayan Wang, et al. A generative vision model that trains with high data efficiency and breaks text-based CAPTCHAs. *Science*, 358(6368):eaag2612, 2017.

[18] Philippe Golle. Machine learning attacks against the ASIRRA CAPTCHA. In *Proceedings of the 15th ACM conference on Computer and communications security*, pages 535–542, 2008.

[19] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[20] Google. Google reCAPTCHA website. https://developers.google.com/recaptcha/intro.

[21] Rich Gossweiler, Maryam Kamvar, and Shumeet Baluja. What's up CAPTCHA? A CAPTCHA based on image orientation. In *Proceedings of the 18th international conference on World wide web*, pages 841–850, 2009.

[22] Gaurav Goswami, Brian M Powell, Mayank Vatsa, Richa Singh, and Afzel Noore. FaceDCAPTCHA: Face detection based color image CAPTCHA. *Future Generation Computer Systems*, 31:59–68, 2014.

[23] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[24] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018.

[25] Drew A Hudson and Christopher D Manning. Compositional attention networks for machine reasoning. *arXiv preprint arXiv:1803.03067*, 2018.

[26] Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2901–2910, 2017.

[27] Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Judy Hoffman, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Inferring and executing programs for visual reasoning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2989–2998, 2017.

[28] Diederik P Kingma and Jimmy Ba. ADAM: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[29] Qiujie Li. A computer vision attack on the ARTiFACIAL CAPTCHA. *Multimedia Tools and Applications*, 74(13):4583–4597, 2015.

[30] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. SSD: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[31] Mateusz Malinowski and Mario Fritz. A multi-world approach to question answering about real-world scenes based on uncertain input. In *Advances in neural information processing systems*, pages 1682–1690, 2014.

[32] Greg Mori and Jitendra Malik. Recognizing objects in adversarial clutter: Breaking a visual CAPTCHA. In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, volume 1, pages I–I. IEEE, 2003.

[33] Hossein Nejati, Ngai-Man Cheung, Ricardo Sosa, and Dawn CI Koh. DeepCAPTCHA: an image CAPTCHA based on depth perception. In *Proceedings of the 5th ACM multimedia systems conference*, pages 81–90, 2014.

[34] NetEase. NetEase's reasoning puzzle CAPTCHA. https://dun.163.com/trial/inference.

[35] NetEase. NetEase's visual reasoning CAPTCHA. https://dun.163.com/trial/space-inference.

[36] NetEase. NetEase's word-order click CAPTCHA. https://dun.163.com/trial/word-order.

[37] Margarita Osadchy, Julio Hernandez-Castro, Stuart Gibson, Orr Dunkelman, and Daniel Pérez-Cabo. No bot expects the DeepCAPTCHA! Introducing immutable adversarial examples, with applications to CAPTCHA generation. *IEEE Transactions on Information Forensics and Security*, 12(11):2640–2653, 2017.

[38] Joseph Redmon and Ali Farhadi. YOLOv3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[39] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[40] Yong Rui and Zicheng Liu. ARTiFACIAL: Automated reverse turing test using facial features. *Multimedia Systems*, 9(6):493–502, 2004.

[41] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11):2673–2681, 1997.

[42] Chenghui Shi, Xiaogang Xu, Shouling Ji, Kai Bu, Jianhai Chen, Raheem Beyah, and Ting Wang. Adversarial CAPTCHAs. *arXiv preprint arXiv:1901.01107*, 2019.

[43] Jiaxin Shi, Hanwang Zhang, and Juanzi Li. Explainable and explicit visual reasoning over scene graphs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8376–8384, 2019.

[44] Suphannee Sivakorn, Iasonas Polakis, and Angelos D Keromytis. I am robot:(deep) learning to break semantic image CAPTCHAs. In *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 388–403. IEEE, 2016.

[45] Mengyun Tang, Haichang Gao, Yang Zhang, Yi Liu, Ping Zhang, and Ping Wang. Research on deep learning techniques in breaking text-based CAPTCHAs and designing image-based CAPTCHA. *IEEE Transactions on Information Forensics and Security*, 13(10):2522–2537, 2018.

[46] Tencent. Tencent waterproof wall website. https://007.qq.com/online.html.

[47] Sheng Tian and Tao Xiong. A Generic Solver Combining Unsupervised Learning and Representation Learning for Breaking Text-Based CAPTCHAs. In *Proceedings of The Web Conference 2020*, pages 860–871, 2020.

[48] Erkam Uzun, Simon Pak Ho Chung, Irfan Essa, and Wenke Lee. rtCAPTCHA: A Real-Time CAPTCHA Based Liveness Detection System. In *NDSS*, 2018.

[49] Shardul Vikram, Yinan Fan, and Guofei Gu. SEMAGE: a new image-based two-factor CAPTCHA. In *Proceedings of the 27th Annual Computer Security Applications Conference*, pages 237–246, 2011.

[50] Luis Von Ahn, Manuel Blum, and John Langford. Telling humans and computers apart automatically. *Communications of the ACM*, 47(2):56–60, 2004.

[51] Luis Von Ahn, Benjamin Maurer, Colin McMillen, David Abraham, and Manuel Blum. reCAPTCHA: Human-based character recognition via web security measures. *Science*, 321(5895):1465–1468, 2008.

[52] Haipeng Wang, Feng Zheng, Zhuoming Chen, Yi Lu, Jing Gao, and Renjia Wei. A CAPTCHA Design Based on Visual Reasoning. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1967–1971. IEEE, 2018.

[53] Ping Wang, Haichang Gao, Qingxun Rao, Sainan Luo, Zhongni Yuan, and Ziyu Shi. A Security Analysis of CAPTCHAs with Large Character Sets. *IEEE Transactions on Dependable and Secure Computing*, 2020.

[54] Xinhua. Xinhua Dictionary website. http://xh.5156edu.com/.

[55] Jeff Yan and Ahmad Salah El Ahmad. Breaking visual CAPTCHAs with naive pattern recognition algorithms. In *Twenty-Third Annual Computer Security Applications Conference (ACSAC 2007)*, pages 279–291. IEEE, 2007.

[56] Jeff Yan and Ahmad Salah El Ahmad. A Low-cost Attack on a Microsoft CAPTCHA. In *Proceedings of the 15th ACM conference on Computer and communications security*, pages 543–554, 2008.

[57] Guixin Ye, Zhanyong Tang, Dingyi Fang, Zhanxing Zhu, Yansong Feng, Pengfei Xu, Xiaojiang Chen, and Zheng Wang. Yet another text CAPTCHA solver: A generative adversarial network based approach. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pages 332–348, 2018.

[58] Binbin Zhao, Haiqin Weng, Shouling Ji, Jianhai Chen, Ting Wang, Qinming He, and Reheem Beyah. Towards evaluating the security of real-world deployed image CAPTCHAs. In *Proceedings of the 11th ACM Workshop on Artificial Intelligence and Security*, pages 85–96, 2018.

[59] Bin B Zhu, Jeff Yan, Qiujie Li, Chao Yang, Jia Liu, Ning Xu, Meng Yi, and Kaiwei Cai. Attacks and design of image recognition CAPTCHAs. In *Proceedings of the 17th ACM conference on Computer and communications security*, pages 187–200, 2010.

[60] Yang Zi, Haichang Gao, Zhouhang Cheng, and Yi Liu. An End-to-End Attack on Text CAPTCHAs. *IEEE Transactions on Information Forensics and Security*, 15:753–766, 2019.