Counteracting Dark Web Text-Based CAPTCHA with Generative Adversarial Learning for Proactive Cyber Threat Intelligence

NING ZHANG, University of Arizona, USA
MOHAMMADREZA EBRAHIMI, University of South Florida, USA
WEIFENG LI, University of Georgia, USA
HSINCHUN CHEN, University of Arizona, USA

Automated monitoring of dark web (DW) platforms on a large scale is the first step toward developing proactive Cyber Threat Intelligence (CTI). While there are efficient methods for collecting data from the surface web, large-scale dark web data collection is often hindered by anti-crawling measures. In particular, text-based CAPTCHA serves as the most prevalent and prohibiting type of these measures in the dark web. Text-based CAPTCHA identifies and blocks automated crawlers by forcing the user to enter a combination of hard-to-recognize alphanumeric characters. In the dark web, CAPTCHA images are meticulously designed with additional background noise and variable character length to prevent automated CAPTCHA breaking. Existing automated CAPTCHA breaking methods have difficulties in overcoming these dark web challenges. As such, solving dark web text-based CAPTCHA has been relying heavily on human involvement, which is labor-intensive and time-consuming. In this study, we propose a novel framework for automated breaking of dark web CAPTCHA to facilitate dark web data collection. This framework encompasses a novel generative method to recognize dark web text-based CAPTCHA with noisy background and variable character length. To eliminate the need for human involvement, the proposed framework utilizes Generative Adversarial Network (GAN) to counteract dark web background noise and leverages an enhanced character segmentation algorithm to handle CAPTCHA images with variable character length. Our proposed framework, DW-GAN, was systematically evaluated on multiple dark web CAPTCHA testbeds. DW-GAN significantly outperformed the state-of-the-art benchmark methods on all datasets, achieving over 94.4% success rate on a carefully collected real-world dark web dataset. We further conducted a case study on an emergent Dark Net Marketplace (DNM) to demonstrate that DW-GAN eliminated human involvement by automatically solving CAPTCHA challenges with no more than three attempts. Our research enables the CTI community to develop advanced, large-scale dark web monitoring. We make DW-GAN code available to the community as an open-source tool in GitHub.

CCS Concepts: • **Security and privacy**; • **Information systems, Web searching and information discovery**;

This material is based upon work supported by the National Science Foundation (NSF) under the following grants: SaTC-1936370, CICI-1917117, and SFS-1921485.

Authors' addresses: N. Zhang (corresponding authors) and H. Chen, University of Arizona, 1130 E. Helen St., Tucson, AZ, 85712; emails: zhangning@email.arizona.edu, hsinchun@arizona.edu; M. Ebrahimi (corresponding authors), University of South Florida, 4202 E. Fowler Avenue, Tampa, FL, 33620; email: ebrahimim@usf.edu; W. Li, University of Georgia, 630 S. Lumpkin St., Athens, GA, 30602; email: weifeng.li@uga.edu.

2158-656X/2022/03-ART21 $15.00
https://doi.org/10.1145/3505226

Additional Key Words and Phrases: Automated CAPTCHA breaking, dark web, generative adversarial networks

ACM Reference format:
Ning Zhang, Mohammadreza Ebrahimi, Weifeng Li, and Hsinchun Chen. 2022. Counteracting Dark Web Text-Based CAPTCHA with Generative Adversarial Learning for Proactive Cyber Threat Intelligence. *ACM Trans. Manage. Inf. Syst.* 13, 2, Article 21 (March 2022), 21 pages.
https://doi.org/10.1145/3505226

1 INTRODUCTION

Illicit platforms have become increasingly common on the dark web [2]. Examples include **Dark Net Markets** (**DNMs**) and carding shops enabling hackers to exchange stolen data, hacking tools, and other illegal products. These malicious products could be detrimental to cyber defense and cause significant loss and damage to cyber infrastructure. To address this issue, proactive **Cyber Threat Intelligence** (**CTI**) aims at monitoring the dark web to inform cybersecurity decision making [6, 8, 9, 19, 28, 33]. For instance, cybersecurity firms such as FireEye identify threats to customer assets from the dark web [14, 18]. Also, with the prevalence of financial data breaches in carding shops and DNMs, monitoring the dark web helps financial institutions alert their customers for potential risks [31]. While developing proactive CTI necessitates automated web crawling for dark web data collection, the dark web extensively employs anti-crawling measures to prevent automated data collection. Text-based CAPTCHA is often known as the most common and difficult anti-crawling measure to counteract in the dark web [5, 34],where CAPTCHA stands for Completely Automated Public Turing Test to tell Computers and Human Apart. Text-based CAPTCHA identifies and blocks web crawlers by presenting a heavily obfuscated image of characters and testing the user's ability to identify the characters shown in the CAPTCHA image. The obfuscation generally involves distorting characters in terms of their font, color, and rotation.

Text-based CAPTCHA can significantly hamper the automation of large-scale dark web collection. When navigating through dark web platforms, crawlers are frequently disrupted by text-based CAPTCHA challenges, which then requires human involvement and is thus detrimental to automated large-scale dark web collection. While recent **machine learning** (**ML**) methods have been developed and shown promising results in automated CAPTCHA breaking [10–12, 15, 37], dark web platforms increase the difficulty of algorithmic CAPTCHA breaking. Text-based CAPTCHA images in the dark web are different from those on regular platforms in two ways. First, CAPTCHA backgrounds are made particularly noisy by adding distracting background consisting of colorful curves and dots. Such noisy backgrounds are challenging for ML-based methods because they need to learn and distinguish a large number of background patterns, in addition to recognizing the characters. Second, while the character length of text-based CAPTCHA images is a key configuration parameter for training ML-based CAPTCHA breaking methods, dark web platforms rarely use CAPTCHA images with a fixed character length. Pre-trained ML methods often have difficulty in breaking CAPTCHA images with different lengths from what they are trained on. In addition to these challenges, there is a lack of human-labeled dark web-specific text-based CAPTCHA datasets for training ML-based CAPTCHA breaking methods. As such, existing CAPTCHA breaking methods often face trouble with effectively facilitating the automated collection of dark web content for CTI.

Motivated by these challenges, our study contributes to the cybersecurity literature by proposing a novel CAPTCHA breaking framework that leverages the state-of-the-art deep learning techniques for breaking dark web CAPTCHA. The proposed framework, named DW-GAN, comprises three sequential components. First, the automated background removal component seeks to remove the dark web noisy background to improve CAPTCHA breaking performance. To this end, we propose a **Generative Adversarial Network** (**GAN**) model, CAPTCHA GAN, that learns to generate CAPTCHA images with relatively clean background from the original patterns with noisy backgrounds. Instead of using millions of labeled CAPTCHA images for training, our CAPTCHA GAN can generate training data to address the lack of labeled training data. Second, the character segmentation component addresses the challenge of variable character length by extracting image segments, each of which contains one single character. Additionally, we extend the core image segmentation technique with a region enlargement procedure for achieving better segmentation. Lastly, the character recognition component detects the character in each CAPTCHA image segment. We utilize the state-of-the-art **Convolutional Neural Network** (**CNN**) to recognize segmented characters.

Our proposed framework was rigorously evaluated on a research testbed comprising various CAPTCHA images from the dark web. Our evaluation experiments show that the proposed framework consistently outperformed the state-of-the-art baseline CAPTCHA-breaking methods. In particular, our ablation analysis shows that our method was able to effectively remove the colorful curves and dots in the background. Moreover, our proposed CAPTCHA segmentation was able to further improve the success rate of breaking CAPTCHA with variable character length over the state-of-the-art baseline techniques. We demonstrate the applicability of our proposed framework through a case study on dark web data collection, where we incorporated our framework into a dark web crawler. Equipped with **Dark Web-GAN** (**DW-GAN**), the crawler successfully collected a medium-sized Dark Net Marketplace within almost five hours without any human involvement, where DW-GAN was able to break CAPTCHA images with no more than three attempts.

The remainder of the article is organized as follows. Section 2 presents a review of related literature to motivate our research and provide a technical background of CAPTCHA-breaking. Section 3 details our proposed design of the DW-GAN framework. Section 4 systematically examines the effectiveness of our proposed framework and its major novelties in comparison with the state-of-the-art benchmark techniques. Section 5 demonstrates the applicability of our proposed framework through a case study on dark web collection. Lastly, Section 6 summarizes our contribution and discusses our future directions.

2 LITERATURE REVIEW

Based on our research objectives, we review two major areas of research. First, we examine dark web data collection for proactive CTI as the major application domain of our study and investigate CAPTCHA as a major challenge hampering dark web data collection. Second, we review the state-of-the-art text-based CAPTCHA breaking methods in prior research. In particular, we explore image preprocessing and background denoising techniques as necessary steps for breaking the dark web CAPTCHA with complex backgrounds. Subsequently, we examine character segmentation as a vital step to breaking CAPTCHA with variable character length. Lastly, we investigate character recognition techniques for distinguishing segmented CAPTCHA characters.

2.1 Dark Web Data Collection for CTI

Cyber attacks are projected to cost the global economy $6 trillion by 2021 [20]. The mitigation of cyber-attacks increasingly relies on gathering hacker intelligence from the dark web [2] to proactively gain reconnaissance on emerging cyber threats and key threat actors [7]. The dark web features a conglomerate of covert illegal platforms, including shops selling stolen credit/debit card and dark net marketplaces facilitating transactions between cybercriminals. Accordingly, dark web data collection has been considered as a key step in developing proactive CTI [4, 26].

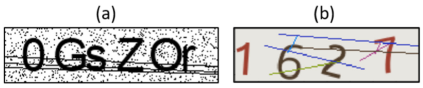


Fig. 1. Examples of CAPTCHA with background and foreground security measures: (a) dots+curves+font size noise (b) curves + foreground noise.

Automated data collection from the dark web is crucial to proactively respond to cyber threats and data breach incidents. However, the automation of dark web data collection is complicated by anti-crawling measures widely adopted in the dark web [27]. These measures include user authentication, session timeout, deployment of cookies, and CAPTCHA. While most of these measures can be effectively circumvented through implementing automated countermeasures in a crawler program, CAPTCHA is the most hampering anti-crawling measure in the dark web that cannot be easily circumvented due to high cognitive capabilities that are often not possessed by automation tools [5, 38]. There are four major types of CAPTCHA: text-based, image-based, video-based, and audio-based. Text-Based CAPTCHA requires subjects to recognize the characters shown in a deliberately obfuscated alphanumeric pattern. Image-based CAPTCHA asks subjects to perform certain actions (e.g., drag and drop) on a specific portion of a given image. Video-based CAPTCHA challenges subjects to choose an option that best describes the content of a given video. Finally, audio-based CAPTCHA plays an audio and requires users to enter the characters mentioned in the audio. Text-based CAPTCHA is the most prevalent type among various types of CAPTCHA in the dark web, and is the focus of our research [35]. We note that despite the ubiquity of traditional text-based CAPTCHA in online space, dark web CAPTCHA patterns are uniquely positioned due to the use of more complicated background noise. Accordingly, while this study is mainly focused on dark-web CAPTCHA as a more challenging problem, the proposed method in this study is expected to be applicable to other types of CAPTCHA without loss of generality.

2.2 Text-based CAPTCHA Breaking Methods

Automated breaking of text-based CAPTCHA is non-trivial due to two main challenges: complex security measures and variable character length. The former involves intentionally obfuscating patterns added to the CAPTCHA image for complicating the recognition task [3]. The latter draws upon the fact that most automated methods trained to break CAPTCHA with a specific length are poorly generalizable to CAPTCHAs with different character length [37]. There are two categories of security measures: foreground security measures and background security measures. Foreground security measures are mainly employed to prevent characters from having a uniform appearance. These measures include font change (i.e., varying the typeface of the character to non-standard unseen fonts), character rotation (i.e., varying the angular position of each character), and color change (i.e., varying the foreground color of each character). Background security measures provide additional obfuscation to the background of the CAPTCHA image by introducing dot noise (i.e., dot-shaped noise applied to the background surface while interfering with the characters), curve noise (i.e., irregular curvatures that cross characters), and color change (i.e., varying the color of noise so that it is not simply removable by elimination of a specific color). Figure 1 illustrates examples of background and foreground security measures.

Extant CAPTCHA-breaking frameworks often employ three steps to address these security measures [35]:

— Image Preprocessing: Performs a series of computer vision processing techniques to remove background and foreground noise.

— Character Segmentation: Separates characters in the CAPTCA image for character-level CAPTCHA breaking (necessary to handle patterns with variable character length).
— Character Recognition: Identifies foreground characters via deep learning-based classification architectures such as CNN.

We summarize prior CAPTCHA-breaking studies with focus on the methods employed at each of these steps in Table 1.

2.2.1 Image Processing and Background Denoising. Preprocessing CAPTCHA images involves applying computer vision techniques to enhance the foreground and denoise the background. Past studies utilize five major image preprocessing techniques for this purpose: normalization, grayscale conversion, Gaussian smoothing, dilation, and erosion. Normalization scales pixel values into a certain range to enhance inconspicuous features [29]. Grayscale conversion changes colored images into grayscale to reduce the negative impact of a high variance in colors [11]. Gaussian smoothing applies Gaussian function to remove details and reduce the impact of curves and dots on the detection of characters' edges [35]. Dilation enlarges shapes proportionally to help repair the missing parts in characters [10]. Finally, erosion down-scales shapes proportionally in order to shrink dot noises [10]. Table 2 summarizes the main strengths of each image preprocessing technique in dealing with background noise. While these techniques help enhance foreground characters, background denoising in complex patterns still remains a challenge [11]. As shown in Table 2, the removal of curve noise is a difficult task that is hard to address with common image preprocessing methods. This is mainly due to the resemblance of the curves to the foreground characters. Accordingly, attempts to remove curves may result in unintended removal of foreground characters. Background denoising is even more difficult for dark web CAPTCHA with complex and noisy backgrounds.

While deep learning models have demonstrated promising results in image processing and can potentially improve background removal, most deep learning models are required to be trained on a large number of unseen background patterns. However, there lacks enough training data of background patterns for training such models. This scarcity is even more severe for specific domains such as the dark web. Recently, Ye et al. [37] have shown that GAN can address this issue by automatically generating background patterns with eliminated curve noise. GAN for background removal comprises two competing neural networks known as generator and discriminator. The generator tries to generate patterns with clean background from input CAPTCHAs. The discriminator tries to identify if the generator has fully removed the background. Nevertheless, the approach in [37] operates at the image level, and thus is not applicable to dark web CAPTCHA with variable length.

2.2.2 Character Segmentation to Address Variable Character Length. Despite the noisy background, another unique challenge of dark web CAPTCHA is the variable character length. Many automated CAPTCHA breaking solutions operate at the image level, where the CAPTCHA image is analyzed in its entirety and the CAPTCHA breaker predicts characters in the CAPTCHA altogether. As such, these solutions need to be trained on fixed-length CAPTCHA. The variable character length makes such image-level CAPTCHA breakers ineffective for two major reasons. First, image-level models that are trained on a pre-specified length are not applicable to CAPTCHAs with a different length of characters. Second, at the image level, the number of class labels grows exponentially with the number of characters (e.g., 10^3 for 3-digit and 10^4 for 4-digit numerical CAPTCHA). As such, a very large number of CAPTCHA images is required for model training because each class label needs to have a sufficient number of training CAPTCHA images [37].

Table 1. Selected Major ML-based CAPTCHA Breaking Studies

Author	Focus	Testbed	Targeted Security Measure		Image Processing	Character Segmentation	Character Recognition
			Foreground	Background			
Nouri and Rezaei [22]	Utilize CNN to break synthesized CAPTCHA for vulnerability study	500,000 randomly synthesized CAPTCHA	Font, rotation, color	Noise, curve, color	Grayscale Conversion, Gaussian Smoothing	No	CNN
Wu et al. [35]	CAPTCHA breaking for numerous Chinese characters	Chinese National Enterprise Credit Information Publicity System (10k)	Font, rotation, color	Noise,color	Grayscale Conversion, Gaussian Smoothing	Yes	CNN
Ferreira et al. [10]	Apply sparsity constraint in CNN to improve accuracy	Kaggle CAPTCHA dataset (87,047 images)	Font, rotation	-	Dilation, Erosion	Yes	CNN
Ye et al. [37]	Use generative adversarial network to remove security features	33 Captcha sets (MS, eBay, Sina, etc.)	Font, rotation, color	Noise, curve, color	GAN-Based Background Removing	No	CNN
Tang et al. [30]	Leverage segmentation and deep learning to break English and Chinese CAPTCHA	11 Captcha sets	Font, rotation, color	Color	Grayscale Conversion, Gaussian Smoothing	Yes	CNN
George et al. [12]	Design a structured probabilistic graphical model for CAPTCHA breaking	4 surface web CAPTCHA sets (reCAPTCHA, Botdetect, Yahoo etc.)	Font, rotation, color	Noise, curve, color	-	Yes	RCN
Le et al. [17]	Utilize synthetic data to train a deep learning model to break CAPTCHA	7 platforms (Baidu, eBay, Yahoo)	Font, rotation	-	-	No	CNN and RNN
Hussain et al. [15]	Leverage segmentation and deep learning to break the CAPTCHA with overlapping characters	Taobao, eBay, and MSN (3,500 CAPTCHA images)	Font, rotation	-	Normalization, Grayscale Conversion	Yes	ANN
Gao et al. [11]	Design a CNN-based solution to break hollow CAPTCHA by applying CFS segmentation method.	Yahoo (1,500 CAPTCHA images)	Font, rotation	Curve, color	Normalization	Yes	ANN

Note: ANN: Artificial Neural Network; CNN: Convolutional Neural Network; GAN: Generative Adversarial Network; CFS: Color Filling Segmentation; RNN: Recurrent Neural Network; RCN: Recursive Cortical Network.

Table 2. Image Processing Techniques to Address Background Security Features in CAPTCHA

Method	Major Goal	Targeted Background Security Measure		
		Dot Noise	Curve Noise	Color
Normalization [15]	Enhance inconspicuous features in images.	×	×	✓
Grayscale Conversion [30]	Omit the impact of different colors.	×	×	✓
Gaussian Smoothing [35]	Lower the impact of dot noise on the character edges.	✓	×	×
Dilation [29]	Fix the missing parts of characters.	×	×	×
Erosion [29]	Shrink the dots' size as much as possible.	✓	×	×

Note: ANN: Artificial Neural Network; CNN: Convolutional Neural Network; GAN: Generative Adversarial Network; CFS: Color Filling Segmentation; RNN: Recurrent Neural Network; RCN: Recursive Cortical Network.

Table 3. Character Segmentation Methods for CAPTCHA Breaking

Method	Major Goal	Source	Targeted Background Security Measure		
			Dot Noise	Curve Noise	Color
CFS	Fills hollow shapes with a different color to differentiate characters.	Gao et al. [11]	✓	×	✓
Interval-based Segmentation	Split CAPTCHA images by fixed intervals.	Tang et al. [30]	×	✓	×
Pixel Distribution	Crops the characters based on the variance of the pixel values.	Hussain et al. [15]	✓	×	×
Contour Detection	Identifies an area containing one single character based on the contour features of the character.	Ferreira et al. [10]	✓	✓	✓

Note: ANN: Artificial Neural Network; CNN: Convolutional Neural Network; GAN: Generative Adversarial Network; CFS: Color Filling Segmentation; RNN: Recurrent Neural Network; RCN: Recursive Cortical Network.

In contrast, character-level CAPTCHA breakers require a much smaller number of class labels (e.g., 10 different classes (i.e., 0, . . . , 9) for numerical patterns). In particular, character-level CAPTCHA breakers perform character segmentation to separate characters from other characters prior to recognition [35]. Four segmentation methods are commonly leveraged in CAPTCHA breaking research: **color filling segmentation** (**CFS**), interval-based, pixel distribution-based, and contour detection-based. We provide a review of these methods in Table 3. As observed in the table, the contour detection method is more suitable for the dark web CAPTCHA since the method can operate despite the changes in font, color, and rotation of the characters. This is mainly due to the independence of the contours from these security measures.

2.2.3 Character Recognition. After identifying the character boundaries via segmentation, the last step involves correctly recognizing the characters within each boundary [11]. As shown in Table 1, among the past studies, CNNs have been widely used for character recognition task [3, 34]. CNNs have shown promising results in counteracting foreground security measures such as rotation, color change, and font change of the characters. A CNN is built upon three main components: convolution layer, sampling layer, and fully connected layer. Each CNN component contributes to a certain aspect of CAPTCHA Character recognition. Convolution layer extracts geometrical salient features from local regions of the input image. For CAPTCHA breaking, this layer extracts rotation-invariant features from characters. The sampling layer combines features from local regions to generate more abstract features. For CAPTCHA breaking, this layer helps identify characters despite differences in their font and sizes. The fully connected layer weights the extracted features and assigns a probability to the output. For CAPTCHA breaking, this layer predicts the pattern's class label based on the extracted features. Accordingly, we expect that CNN can effectively contribute to dark web CAPTCHA character recognition after proper character segmentation.

2.3 Research Gaps and Questions

Existing CAPTCHA breaking methods are not designed for addressing specific characteristics of the dark web CAPTCHA. As such, security analysts often face trouble with effectively facilitating the automated collection of dark web content for CTI. Specifically, two major research gaps are identified from the review of prior studies. First, there lack studies offering methods for breaking CAPTCHA with complex noisy backgrounds in the dark web. Second, existing methods do not address CAPTCHA with variable character lengths and noisy backgrounds within a unified framework. Based on these gaps, we pose the following research question: *How can we design an automated CAPTCHA breaking framework to address the noisy background and variable character length in the dark web?*

3 RESEARCH DESIGN

To address our research question, we propose DW-GAN, a novel GAN-based framework that utilizes background denoising, character segmentation, and character recognition to automatically break dark web CAPTCHA. DW-GAN was implemented using PyTorch and OpenCV open-source libraries. DW-GAN aims at counteracting background security measures and address the challenge of variable character length for dark web CAPTCHA. Consistent with the steps described in the literature review, DW-GAN comprises three major components shown in Figure 2.

As illustrated in the figure, first, the GAN-based background denoising component removes the noisy background from a given dark web CAPTCHA image. Then, the denoised CAPTCHA is processed using Gaussian smoothing for removing the residual noise. As opposed to the GAN approach used by Ye et al. [37] that operates at the image level, and thus is not applicable to dark web CAPTCHA with variable length, our proposed DW-GAN is designed to recognize CAPTCHA at the character level, which does not depend on the CAPTCHA length. Once the background is denoised, characters are segmented using an enhanced contour detection algorithm so that the remaining steps will not depend on the character length of the CAPTCHA image. To this end, we enhance the core border tracing algorithm with a subsequent region enlargement procedure, which is described later. Lastly, segments of the original CAPTCHA image are fed to a CNN for deep learning-based character recognition. We detail each component of DW-GAN in the following subsections.

3.1 Background Denoising

3.1.1 Background Denoising: CAPTCHA GAN. Attaining a training dataset encompassing all possible background patterns might not be practical for dark web CAPTCHA as it requires expensive human-labeled data and excessive human involvement. As such, we propose to leverage GAN to automatically generate CAPTCHA background patterns for training background denoising.

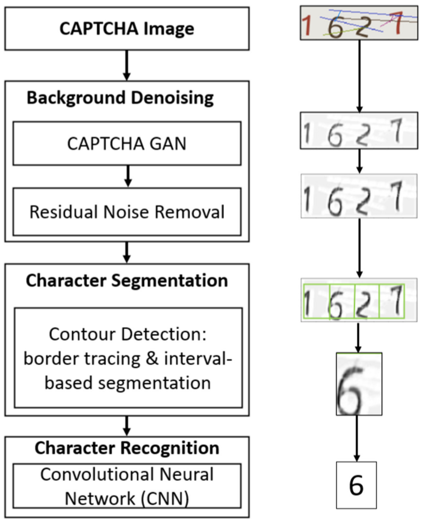


Fig. 2. DW-GAN framework for breaking dark web-specific CAPTCHA images (left) and corresponding illustration (right).

Specifically, the background denoising component in DW-GAN consists of two sub-components: (1) the CAPTCHA GAN sub-component, a customized GAN architecture to automatically learn how to remove background noise, and (2) the residual background noise removal sub-component to further enhance the CAPTCHA foreground. CAPTCHA GAN aims at reducing various background curve noises in CAPTCHA images automatically. This is achieved by incentivizing the CAPTCHA GAN generator to generate background noise-free counterparts of original CAPTCHA images. This generative feature of CAPTCHA GAN allows for training the model with only a small labeled dataset (i.e., 500 dark web CAPTCHA images) and achieving a high performance in background denoising. The learning process for background removal in CAPTCHA GAN includes four major steps:

— Step 1: The generator seeks to create background noise-free CAPTCHA image x from the original dark web CAPTCHA image y.
— Step 2: The generated pattern and the corresponding original CAPTCHA image are fed to the discriminator to assess whether the background noise has been completely removed.
— Step 3: The generator improves by learning via minimizing the loss function $\mathcal{L}_{\mathcal{G}}$, a pixel-to-pixel cross-entropy loss that compares the generator's output and the noise-free CAPTCHA image x. As in the conventional GAN [13], the loss minimization is conducted via gradient descent as part of the back propagation in the generator's neural network parameterized by network weights θ_g. The generator's loss function is given by $\mathcal{L}_{\mathcal{G}} = \nabla_{\theta_g} \frac{1}{N} \sum_n^N [\log(1 - D(G(y_n)))]$.
— Step 4: The discriminator improves by maximizing the loss function $\mathcal{L}_{\mathcal{D}}$ that compares the true label (i.e., incomplete (0) vs. complete background removal (1)) and the discriminators' output. the loss minimization is conducted via gradient ascent as part of the back propagation in the discriminator's neural network parameterized by network weights θ_d. The discriminators' loss is given by $\mathcal{L}_{\mathcal{D}} = \nabla_{\theta_d} \frac{1}{N} \sum_n^N [(\log(D(x_n))) + \log(1 - D(G(y_n)))]$. to make the discrimination more accurate even on the denoised CAPTCHA image from generator's output.

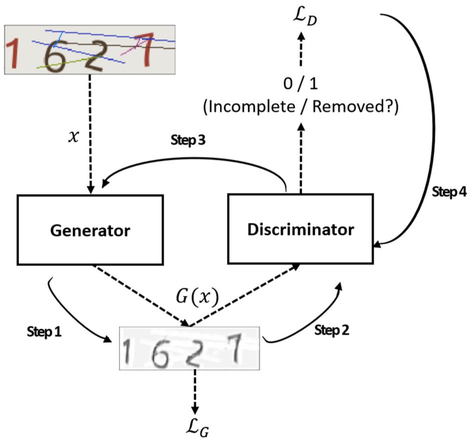


Fig. 3. Abstract view of CAPTCHA GAN for background denoising.

Steps 1–4 are repeated until the generator and discriminator reach the equilibrium condition, where neither is able to improve the performance further [13]. Figure 3 depicts the abstract view of these steps.

CAPTCHA GAN's process in Figure 3 can be formalized as the minimax game with value function V in Equation (1), in which $G(x_n)$ and $D(x_n)$ denote the generator and discriminator networks, respectively.

$$\min_G \max_D V(D, G) = \frac{1}{N} \sum_n^N [(\log(D(x_n))) + \log(1 - D(G(y_n)))]. \tag{1}$$

N denotes the number of images in the dataset and x_n represents the input CAPTCHA image. Also, $G(x_n)$ associates with $\mathcal{L}_G$ and $D(x_n)$ corresponds to discriminator's loss $\mathcal{L}_\mathcal{D}$. CAPTCHA GAN was implemented in Python using PyTorch deep learning library. All training and testing processes for experiments in Section 4 were executed on a single Nvidia RTX 2070 GPU with 2,560 CUDA cores and 8 GB internal memory. To enhance reproducibility, the model specifications of CAPTCHA GAN, including the number of layers, neurons, and activation functions are given in Appendix A. Also, the implementation of CAPTCHA GAN is available at https://github.com/johnnyzn/DW-GAN as part of the DW-GAN's source code.

3.1.2 Background Denoising: Residual Noise Removal. While GAN-based background denoising seeks to remove all curve noise in the background, some residual noise may still remain. This residual noise could potentially interfere with the subsequent segmentation and character recognition. Accordingly, we utilize a combination of three image processing techniques to further remove the remaining background noise. Grayscale conversion is first leveraged to reduce the color variance as a preliminary step that alleviates color noise. Then, Gaussian smoothing is employed to reduce the visibility of the residual background noise by blurring the unnecessary details in the image. Lastly, normalization is applied to distinguish the foreground from the background color. An example of the results of residual noise removal is depicted in Figure 4.

3.2 Character Segmentation

After background denoising, character segmentation is conducted to identify the boundary of characters. As reviewed in Table 3, contour detection is suitable for segmenting characters in dark web CAPTCHA images due to its robustness against the dark web-specific noises. Among various con-

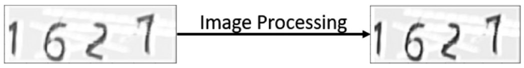


Fig. 4. An example of the result of residual noise removal.

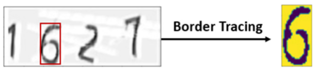


Fig. 5. An example of the result of character segmentation.

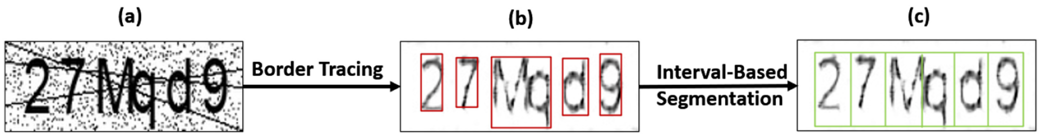


Fig. 6. Our character segmentation process of enhancing border tracing with interval-based segmentation.

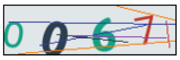
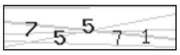
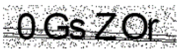
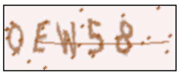
tour detection methods, the *border tracing* algorithm has been successfully used for segmenting CAPTCHA images since the algorithm can effectively identify the boundary pixels of the character region and separate the characters from the background [36]. This is mainly because characters in text-based CAPTCHA often have distinguishable borders. Border tracing algorithm has two main steps [25]. In the first step, the image is converted to binary pixels (black and white) and is scanned from the upper left to the bottom right pixel. In the second step, for each pixel, the algorithm searches a square neighborhood (e.g., 3x3 pixels) to find the direction of the edges and define minimal regions to bound the character. The result of such a segmentation process is illustrated in Figure 5.

While effective, sometimes the detected regions obtained from border tracing might not be large enough to encompass the entire character (e.g., digit “6” in Figure 6(b)). The core border tracing method has a tendency to yield minimal segments that may not include the entire character. As a result, incorrect segmentation would result in the false recognition of the corresponding character, and thereby the entire CAPTCHA image. To address this issue, we enhance border tracing with subsequent region enlargement to encompass the entire character via a two-step process that enhances the initial regions found by the core border tracing algorithm:

(1) The initial character regions are first detected by border tracing algorithm (Figure 6(b)).
(2) Subsequently, the detected regions are overlapped with the maximal regions resulted from dividing the CAPTCHA image into fixed intervals. This could result in detecting a larger region that bounds the character (Figure 6(c)).

As shown in the figure, characters M and q have a considerable overlap that leads to being incorrectly identified as one character by border tracing. However, the subsequent interval-based segmentation identifies them as two imperfect characters (i.e., an incomplete “M” and a “q” with an extra stroke on the left). Given that the subsequent character recognition component is trained on various forms of incomplete characters, there is a reasonable chance that the incomplete “M” and “q” are correctly recognized. Of course, in severe cases of overlapping characters, where it could be also difficult for a human observer to distinguish characters, the character recognition component is more likely to make mistakes.

Table 4. Summary of our Dark Web CAPTCHA Testbed

Category	Source	Amount	Character Length	Character Type	Sample
Dark Web CAPTCHA	**Rescator-1**	500	4	Digit	
	Rescator-2	500	5	Digit	
	Yellow Brick	500	6	Digit+Letter	
Open-Source Synthesizer		11,000 for each character length	4, 5, 6, 7	Digit+Letter	

3.3 Character Recognition

Consistent with prior literature [3, 34], we leverage CNN to detect characters in extracted CAPTCHA segments. Our character recognition CNN stacks convolutional layers and sampling layers to detect characters via the architecture described in Appendix A. The convolutional layer extracts features from local regions of the segmented CAPTCHA image and the sampling layer combines extracted features across multiple local regions to identify fine-grained features (e.g., lines and edges). Another convolutional-sampling structure of such kind is then stacked over to extract features from larger regions through combining lines and edges into more abstract features informative of characters. Such a stacked structure further addresses CAPTCHA security measures such as rotation and font size change by jointly considering features from multiple local regions. The extracted features are then used to detect characters in a fully connected layer. Given the successful CNN training practices in the literature [23, 32], we used Adam optimizer [16] to minimize the cross entropy loss to train the model and utilize **Rectified Linear Units** (**ReLU**) activation function [21] and the dropout mechanism to improve the efficiency of model training.

4 EVALUATION

We systematically evaluated the performance of our proposed framework in breaking CAPTCHA images with a testbed encompassing dark web CAPTCHA datasets. In particular, our CAPTCHA testbed comprised text-based CAPTCHA images from three different dark web datasets and a widely used open-source CAPTCHA synthesizer. The dark web datasets included three sets of CAPTCHA images: two sets from carding shops (Rescator-1 and Rescator-2) and one set from a newly-emerged DNM (Yellow Brick), as shown in Table 4. All three datasets were suggested by CTI experts. These datasets were selected based on their popularity and scale in the dark web. For each dataset, a TOR-routed spider was developed to collect 500 CAPTCHA images. The three datasets were labeled and inspected by two CTI experts. The second data source of our testbed was an open-source CAPTCHA synthesizer. We leveraged the synthesizer to generate CAPTCHA images with controllable security measures that enabled us to create a fair comparison for different methods on CAPTCHA images with variable character length. These datasets along with their CAPTCHA examples are summarized in Table 4.

We conducted three experiments to evaluate the effectiveness of our proposed research design:

— Experiment 1 examined the overall CAPTCHA breaking performance of our proposed framework in comparison with the state-of-the-art baseline methods. This experiment targets the dark web-specific condition of lacking labeled CAPTCHA images for training. To this end, we sampled 500 images from each of the dark web CAPTCHA datasets serving as the evaluation testbed. The CAPATCHA breaking performance was measured by success rate [1, 37], a commonly used metric representing the percentage of CAPTCHA images that are successfully recognized [37]. A successful recognition entails correctly recognizing *all* characters in the CAPTCHA image. Three state-of-the-art methods from past research, were used as benchmarks: Image-level CNN [17], Image-level CNN with preprocessing (grayscale conversion, normalization, and Gaussian smoothing) [22], and Character-level CNN with interval-based segmentation [30].

— Experiment 2 investigated the effect of each component in our proposed DW-GAN framework through an ablation analysis. To this end, the framework's major components, including background denoising, border tracing segmentation, and interval-based segmentation were removed from DW-GAN framework to evaluate their impact on the automated CAPTCHA breaking performance. The ablation analysis was thoroughly conducted on all dark web datasets used in benchmark evaluation in Experiment 1. Also, consistent with Experiment 1, the success rate was adopted as the evaluation criterion.

— Experiment 3 evaluated the efficacy of DW-GAN components to determine whether background denoising and character segmentation improved CAPTCHA breaking performance. Accordingly, this experiment encompassed two sub-experiments as detailed below. We employed the CAPTCHA synthesizer to generate a wide range of background noise with variable lengths.

  – Experiment 3.1 examined the contribution of the background denoising component to the overall performance in DW-GAN. We evaluated the effect of our background denoising component before and after its application to each type of background noise. To measure the effect, we adopted the **structural similarity** (**SSIM**) metirc [24], a widely used metric to measure the similarity of two images. SSIM can be used to determine the extent to which the background security measures are counteracted through our background denoising component. A higher SSIM between the denoised image and its clean version (without background noise) indicates a more effective removal of background security measures.

  – Experiment 3.2 investigated the effect of the character segmentation component on the overall performance. This experiment compared the performance of DW-GAN with a benchmark image-level method and two character-level methods to evaluate how character segmentation improved CAPTCHA breaking performance when character length varied. In accordance with the common dark web CAPTCHA length, the character lengths ranged from 4 to 7. The evaluation was based on a 90%-10% train-test split. That is, our training set contained 50,000 CAPTCHA images (10,000 images for each character length) and the testing set included 5,000 CAPTCHA images (1,000 images for each character length).

4.1 Results of Experiment 1: Benchmark Evaluation

We compared the performance of DW-GAN to the state-of-the-art CAPTCHA breaking methods across the dark web datasets. The success rates for each method are shown in Table 5. The asterisks denote the statistical significance obtained from paired t-test between the results of DW-GAN and the above-mentioned benchmark methods (P-values are significant at 0.05:*, 0.01:**, 0.001:***).

Table 5. Benchmark Evaluation of DW-GAN Against the State-of-the-art Automated CAPTCHA Breaking Methods Across Different CAPTCHA Datasets from the Dark Web

Method	**Dark Web CAPTCHA**		
	Rescator-1	**Rescator-2**	**Yellow Brick**
Image-level CNN only [17]	63.57%***	35.05%***	5.88%***
Image-level CNN + Preprocessing [22]	70.5%**	36.04%***	7.84%***
Character-level CNN + Segmentation [30]	88.12%**	77.23%**	93.72%*
DW-GAN (Ours)	**94.4%**	**97.50%**	**95.98%**

Values in boldface denote the highest performance.

Three notable observations can be made from the results of Experiment 1. First, comparing the automated CAPTCHA breaking method in [17] against [22], it is observed that preprocessing improved the success rate across all datasets. Second, the character-level method proposed by [30] generally achieved a higher success rate compared to image-level CAPTCHA breaking methods. Third, and most importantly, our proposed framework with GAN-based background denoising yielded the highest success rate across all datasets. Overall, DW-GAN outperformed the state-of-the-art benchmark methods with statistically significant margins in dark web CAPTCHA. DW-GAN's performance is attributed to combining background denoising and character recognition in a unified framework. The contribution of each component is further investigated in Experiments 2 and 3.

4.2 Results of Experiment 2: Ablation Analysis

To gauge the contribution of each component in the proposed DW-GAN framework, we further evaluated the CAPTCHA breaking performance (i.e., success rate) by eliminating each major component in DW-GAN. To retain the basic functionality of CAPTCHA breaking, we utilized at least one segmentation method as well as the CNN for character recognition. Accordingly, we eliminated the background denoising component, border tracing segmentation, and interval-based segmentation in a consecutive manner resulting in three alternative models to compare with the proposed DW-GAN. The results of the ablation analysis for these three models across the dark web datasets in Experiment 1 are given in Table 6.

As shown in Table 6, all three essential components of DW-GAN in our design significantly contributed to the performance of automated CAPTCHA breaking on average across all three datasets. Specifically, eliminating the background denoising component resulted in approximately 28% performance loss (67.83% vs. 95.6% in Table 6) on average across the three datasets. Similarly removing border tracing segmentation and interval-based segmentation leads to a 27% and 40% performance drop, respectively (68.21% and 57.07% vs. 95.96%). These results indicate that background denoising, border tracing segmentation, and interval-based segmentation improved the performance significantly. We note that for Rescator2, due to less complicated background, border tracing alone could be sufficient for effective automated CAPTCHA breaking, and thus there is less need for interval-based segmentation. That is, our DW-GAN framework outperformed with a larger margin on more complex backgrounds with various noise types.

4.3 Results of Experiment 3.1: Evaluating Background Denoising

To evaluate the effect of our background denoising component, we first compared the SSIM of original CAPTCHA images with different types of background noise to clear CAPTCHA images (same

Table 6. Ablation Analysis of Each Component in the DW-GAN's Framework

Model	**Dark Web CAPTCHA**			**Average Performance**
	Rescator-1	**Rescator-2**	**Yellow Brick**	
DW-GAN without Background Denoising	93.4%	88.12%	21.96%	67.83%
DW-GAN without Border Tracing Segmentation	77.2%	88.91%	38.82%	68.21%
DW-GAN without Interval-Based Segmentation	63.2%	**98.02**%	10.00%	57.07%
DW-GAN	**94.4%**	97.50%	**95.98**%	**95.96**%

Values in boldface denote the highest performance.

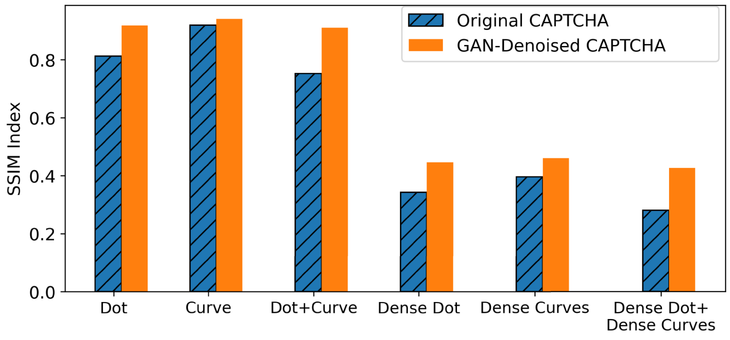


Fig. 7. SSIM Change Before and After Background Denoising for Different Security Measure Combinations.

pattern without any background noise). Then we applied our proposed GAN-based background denoising on the original image and computed the SSIM with respect to the clear CAPTCHA image. The results of these two comparisons are shown in paired bars for each background noise type in Figure 7. As seen in the figure, our GAN-based background denoising method shows consistently higher similarity to the clear image for various types of background noise, including dot, curves, dot+curves, dense dots, dense curves, and dense dot with dense curves.

The high SSIM index achieved by DW-GAN suggests that GAN-based background denoising is capable of removing a wide variety of background noise that are widely used in the dark web.

4.4 Results of Experiment 3.2: Evaluating Character Segmentation

To gauge the effect of our proposed segmentation component in DW-GAN, we applied our segmentation method to various lengths of CAPTCHA images and compared the success rate with other image-level methods as well as common segmentation methods in character-level CAPTCHA breaking. The corresponding success rates are shown in Table 7.

We highlight two major observations from Table 7. First, while the CAPTCHA breaking performance of image-level methods drastically decreased as the character length increased, the variation in character length had a significantly smaller impact on the CAPTCHA breaking performance of our proposed segmentation method in DW-GAN. This shows that DW-GAN's segmentation method performed better than other segmentation methods. Second, consistent with the result of

Table 7. Evaluating the Effect of the Proposed Segmentation Compoent in DW-GAN on CAPTCHA Patterns with Variable Lengths

Character Length	Image-level CNN + Preprocessing	Character-Level + Interval-Based Seg	Character-Level + Border Tracing	DW-GAN's Segmentation
4	56.98%	76.92%	70.07%	79.92%
5	48.43%	73.11%	66.51%	74.35%
6	36.88%	67.81%	65.33%	71.87%
7	29.29%	64.27%	59.68%	69.09%

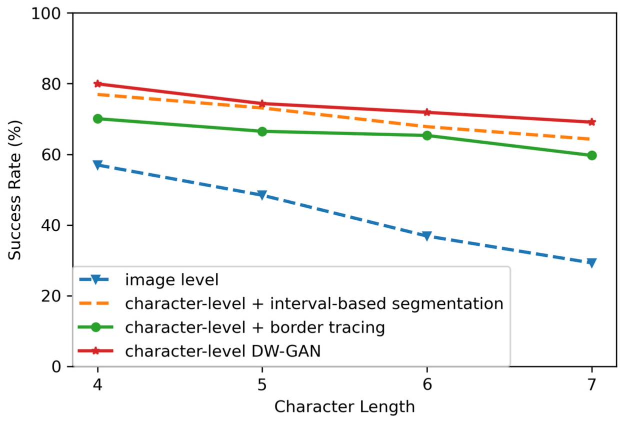


Fig. 8. Sensitivity of success rate to increasing the character length.

Experiment 1, character-level methods demonstrated a higher success rate than image-level methods over all character lengths. To further demonstrate the sensitivity of success rate to increasing the character length for image-level and character-level methods, we plot the performance while increasing the character length in Figure 8. Observing that DW-GAN maintains a high performance as character length increases signifies the effectiveness of the enhanced character segmentation proposed in DW-GAN.

5 DARK WEB CASE STUDY: YELLOW BRICK DNM

We further demonstrate the application of our proposed framework on a real-world DNM. The purpose of this case study is two-fold. First, we sought to show that our proposed framework could detect and break the CAPTCHA to facilitate scalable, automated dark web data collection. Second, we aim at showing that our proposed framework was able to effectively remove human involvement from dark web data collection. Figure 9 illustrates the process of employing DW-GAN for automated dark web CAPTCHA breaking in a DNM.

DW-GAN can be embedded in a dark web crawler. When a CAPTCHA challenge is encountered in a targeted dark web platform, DW-GAN is activated to break the CAPTCHA within a few attempts. The entire process can be broken down into several stages. First, the crawler navigates to the homepage of the targeted DNM. Parsing the login page, the crawler captures the CAPTCHA image and send it to our trained DW-GAN component. DW-GAN denoises, segments, and recognizes the CAPTCHA image. The recognized result would return to our crawler. Finally, the crawler would iteratively enter the predicted result until the correct prediction occurs (i.e., the platform permits access). As such, the DW-GAN enhanced crawler is guaranteed to collect illicit product information without human interaction.

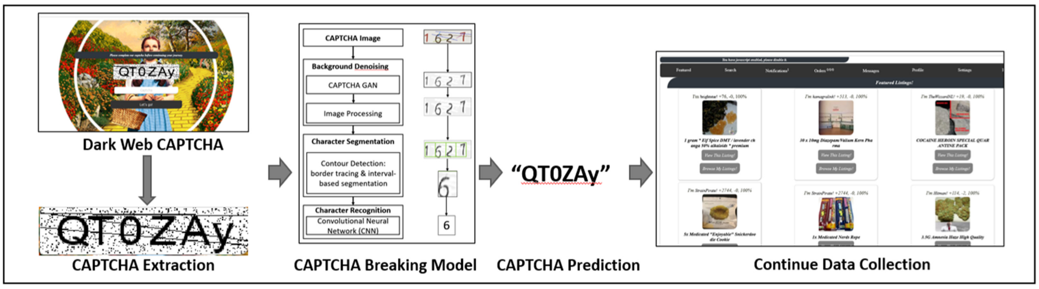


Fig. 9. An automated process for dark web CAPTCHA breaking in a dark net market with DW-GAN.

Based on the above automated process in Figure 9, a Tor-routed web crawler was developed and equipped with DW-GAN to automatically collect illegal advertised product pages from the Yellow Brick DNM. The crawler first detected the login page and automatically entered credentials. Then, the crawler collected all Onion links that contained the information of illegal items by sending HTTP requests to the platform server. Yellow Brick requires re-login and bypassing CAPTCHA for every 15 HTTP requests on average. Without our automated solution, the web crawler needs human interventions every three minutes to manually bypass the CAPTCHA challenges issued by the platform.

Using a crawler enhanced by our DW-GAN, we were able to collect 1,831 illegal products from Yellow Brick. Among these products, there were 286 cybersecurity-related items, including 102 stolen credit cards, 131 stolen accounts, 9 forged document scans, 44 hacking tools, and 1,223 drug-related products (opioid, cocaine, etc.). Overall, collecting Yellow Brick market with DW-GAN took about 5 hours without human involvement. In particular, each HTTP request took 8.8 seconds for loading a new webpage; therefore crawling 1,831 pages took 268.5 minutes. Solving the recurring CAPTCHA challenges (per each 15 HTTP requests) took our DW-GAN crawler 18.6 seconds. Overall, the proposed framework could automatically break CAPTCHA with no more than three attempts. Breaking all CAPTCHA images take about 76 minutes in total for all 1,831 product pages, a process that is fully automated.

6 CONCLUSION AND FUTURE DIRECTIONS

Text-based CAPTCHA breaking has been a major challenge for collecting large-scale dark web data for developing proactive CTI. Two major challenges have been the particularly noisy background and the variable character length of CAPTCHA images. Leveraging GAN and CNN, we propose a novel and automated framework for breaking text-based CAPTCHA in the dark web. The proposed framework utilizes GAN to counteract background security measures for dark web-specific CAPTCHA and leverages an enhanced border tracing algorithm to extract segments encompassing single characters from CAPTCHA images. Our DW-GAN was evaluated on several dark web CAPTCHA datasets. DW-GAN significantly outperformed the state-of-the-art benchmark methods on all datasets in the dark web research context, where there is a lack of labeled CAPTCHA data. Moreover, the proposed framework maintained relatively consistent CAPTCHA breaking performance when character length varies. While collecting dark web data traditionally requires significant human effort, a web crawler equipped with DW-GAN is able to collect large-scale dark web data without human involvement.

Future research is needed to counteract more sophisticated CAPTCHA patterns in the dark web. For instance, we have observed some newly emerged small-sized dark web platforms that complement the text-based CAPTCHA with a "question-answering" challenge. In addition to solving the text-based CAPTCHA, these platforms require user to answer a question. For example, the user may be asked to answer queries such as "adding two numbers" or "the first month of the year." Enhancing DW-GAN for other emerging CAPTCHA challenges remains as a promising future research direction.

APPENDICES

A APPENDIX

This section details the specifications and parameter/hyperparameter settings of DW-GAN. The architecture of DW-GAN was aimed to be parsimonious. To train the deep learning models in DW-GAN, including the generator and discriminator's networks, after a light-weight parameter tuning in a grid search manner, we adopted the learning rate of 0.0035. Also, to utilize the maximum GPU memory, we set the batch size for training process to 200. The GAN-based background denoising was effectively trained after 1,000 epochs. Given that the complexity of character recognition is often less than that of background denoising, effective training the CNN model took 100 epochs. Table 8 includes selected learning rates from the parameter tuning process.

Table 9 summarizes the generator's architecture in our CAPTCHA GAN for background denoising. The generator has 5 convolutional layers with 64, 128, 128, 64, and 1 neural units. As noted, the filter size in convolutional layers was 3×3.

Table 10 includes the specifications of discriminator's architecture in our CAPTCHA GAN. The discriminator encompasses six convolutional layers followed by a fully connected layer. These convolutional layers have 16, 32, 64, 128, 256, and 256 neural units, respectively. Similar to the generator, 3×3 filters were used in the convolutional layers.

Table 8. Different Learning Rates' Impact on DW-GAN Success Rate

Learning Rate	**0.035**	**0.0035**	**0.001**
DW-GAN	59.2%	**94.4%**	86.6%

Table 9. Generator's Parameter Specifications for Background Denoising in CAPTCHA GAN

Layer Number	**Element**	**Parameter**	**Value**
1	Convolutional Filter	[filter number, size, Stride, pad]	[64, 3x3, 1, 1]
2	Convolutional Filter	[filter number, size, Stride, pad]	[128, 3x3, 1, 1]
3	Convolutional Filter	[filter number, size, Stride, pad]	[128, 3x3, 1, 1]
4	Convolutional Filter	[filter number, size, Stride, pad]	[64, 3x3, 1, 1]
5	Convolutional Filter	[filter number, size, Stride, pad]	[1, 3x3, 1, 1]

Table 10. Discriminator's Parameter Specifications for Background Denoising in CAPTCHA GAN

Layer Number	**Element**	**Parameter**	**Value**
1	Convolutional Filter	[filter number, size, Stride, pad]	[16, 3x3, 1, 1]
2	Convolutional Filter	[filter number, size, Stride, pad]	[32, 3x3, 1, 1]
3	Convolutional Filter	[filter number, size, Stride, pad]	[64, 3x3, 1, 1]
4	Convolutional Filter	[filter number, size, Stride, pad]	[128, 3x3, 1, 1]
5	Convolutional Filter	[filter number, size, Stride, pad]	[256, 3x3, 1, 1]
6	Convolutional Filter	[filter number, size, Stride, pad]	[256, 3x3, 1, 1]
7	Fully-Connected	[neuron number]	[1]

Table 11. Character-Level CNN Parameter Specifications for Character Recognition

Layer Number	Element	Parameter	Value
1	Convolutional Filter	[filter number, size, Stride, pad]	[32, 3x3, 1, 1]
	Max Pooling	[size, pad]	[2x2, 2]
2	Convolutional Filter	[filter number, size, Stride, pad]	[64, 3x3, 1, 1]
	Max Pooling	[size, pad]	[2x2, 2]
3	Convolutional Filter	[filter number, size, Stride, pad]	[64, 3x3, 1, 1]
	Max Pooling	[size, pad]	[2x2, 2]
4	Fully-Connected	[neuron number]	[1,024]
5	Fully-Connected	[neuron number]	[256]
6	Fully-Connected	[neuron number]	[character number]

Table 12. Activation Functions' Impact on DW-GAN's Success Rate

Activation Function	ReLU	Tanh	Softmax
DW-GAN	94.4%	74.4%	14.8%

Table 13. The Impact of the Number of Convolutional Layers of the Character Recognition Component on DW-GAN's Success Rate

Convolutional Layer Number	2	3	4
DW-GAN	63.2%	94.4%	90.2%

Table 11 includes the CNN architecture used for the character-level recognition in DW-GAN. The CNN architecture has three convolution layers (each followed by a max-pooling layer) and three fully connected layers. As noted, ReLU was used as activation function.

Since the activation function and the number of convolutional layers have significant effects on the performance of the CNN model, we show selected results of their parameter tuning process to optimize the character recognition. Specifically, for the activation function, we compared ReLU, Tanh, and Softmax (Table 12). ReLU performed the best for the character recognition task in DW-GAN. Also, varying the number of convolutional layers for our CNN model indicated that three convolutional layers perform the best for CAPTCHA breaking. The comparison results are shown in Table 13.

B ONLINE RESOURCES

The implementation of DW-GAN and its corresponding datasets are available at https://github.com/johnnyzn/DW-GAN.

REFERENCES

[1] Elie Bursztein, Matthieu Martin, and John Mitchell. 2011. Text-based CAPTCHA strengths and weaknesses. In *Proceedings of the 18th ACM Conference on Computer and Communications Security*. 125–138.

[2] Hsinchun Chen. 2012. *Dark web: Exploring and Data Mining the Dark Side of the Web*. Springer, New York.

[3] Jun Chen, Xiangyang Luo, Yanqing Guo, Yi Zhang, and Daofu Gong. 2017. A survey on breaking technique of text-based CAPTCHA. *Security and Communication Networks* 2017 (2017), 1–15.

[4] Roger H. L. Chiang, Paulo Goes, and Edward A. Stohr. 2012. Business intelligence and analytics education, and program development: A unique opportunity for the information systems discipline. *ACM Transactions on Management Information Systems* 3, 3, Article 12 (2012), 13 pages. DOI:https://doi.org/10.1145/2361256.2361257

[5] Po-Yi Du, Ning Zhang, Mohammadreza Ebrahimi, Sagar Samtani, Ben Lazarine, Nolan Arnold, Rachael Dunn, Sandeep Suntwal, Guadalupe Angeles, Robert Schweitzer, and Hsinchun Chen. 2018. Identifying, collecting, and presenting hacker community data: Forums, IRC, carding shops, and DNMs. In *Proceedings of the IEEE International Conference on Intelligence and Security Informatics*. IEEE, 70–75.

[6] Mohammadreza Ebrahimi, Yidong Chai, Sagar Samtani, and Hsinchun Chen. Forthcoming. Cross-lingual cybersecurity analytics in the international dark web with adversarial deep representation learning. *MIS Quarterly* (Forthcoming). DOI:https://doi.org/10.25300/MISQ/2022/16618

[7] Mohammadreza Ebrahimi, Jay F. Nunamaker Jr, and Chen Hsinchun. 2020. Semi-supervised cyber threat identification in dark net markets: A transductive and deep learning approach. *Journal of Management Information Systems* 37, 3 (2020), 694–722.

[8] Mohammadreza Ebrahimi, Sagar Samtani, Yidong Chai, and Hsinchun Chen. 2020. Detecting cyber threats in non-english hacker forums: An adversarial cross-lingual knowledge transfer approach. In *Proceedings of the 2020 IEEE Security and Privacy Workshops*. IEEE, 20–26.

[9] Mohammadreza Ebrahimi, Mihai Surdeanu, Sagar Samtani, and Hsinchun Chen. 2018. Detecting cyber threats in non-english dark net markets: A cross-lingual transfer learning approach. In *Proceedings of the IEEE International Conference on Intelligence and Security Informatics*. IEEE, 85–90.

[10] Diogo Daniel Ferreira, Luís Leira, Petya Mihaylova, and Petia Georgieva. 2019. Breaking text-based CAPTCHA with sparse convolutional neural networks. In *Proceedings of the Iberian Conference on Pattern Recognition and Image Analysis*. Springer, 404–415.

[11] Haichang Gao, Wei Wang, Jiao Qi, Xuqin Wang, Xiyang Liu, and Jeff Yan. 2013. The robustness of hollow CAPTCHAs. In *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*. 1075–1086.

[12] Dileep George, Wolfgang Lehrach, Ken Kansky, Miguel Lázaro-Gredilla, Christopher Laan, Bhaskara Marthi, Xinghua Lou, Zhaoshi Meng, Yi Liu, Huayan Wang, Alex Lavin, and D. Scott Phoenix. 2017. A generative vision model that trains with high data efficiency and breaks text-based CAPTCHAs. *Science* 358, 6368 (2017), 1–19. https://www.science.org/doi/10.1126/science.aag2612.

[13] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT press.

[14] Katherine Heires. 2016. Terror tech: New and innovative tools are being deployed in the ongoing fight against terrorism. *Risk Management* 63, 10 (2016), 28–32.

[15] Rafaqat Hussain, Hui Gao, and Riaz Ahmed Shaikh. 2017. Segmentation of connected characters in text-based CAPTCHAs for intelligent character recognition. *Multimedia Tools and Applications* 76, 24 (2017), 25547–25561.

[16] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations*.

[17] Tuan Anh Le, Atilim Günes Baydin, Robert Zinkov, and Frank Wood. 2017. Using synthetic data to train neural networks is model-based reasoning. In *Proceedings of the 2017 International Joint Conference on Neural Networks*. IEEE, 3514–3521.

[18] Shing-Han Li, Yu-Cheng Kao, Zong-Cyuan Zhang, Ying-Ping Chuang, and David C. Yen. 2015. A network behavior-based botnet detection mechanism using PSO and k-means. *ACM Transactions on Management Information Systems* 6, 1, Article 3 (2015), 30 pages. DOI:https://doi.org/10.1145/2676869

[19] Yizhi Liu, Fang Yu Lin, Zara Ahmad-Post, Mohammadreza Ebrahimi, Ning Zhang, James Lee Hu, Jingyu Xin, Weifeng Li, and Hsinchun Chen. 2020. Identifying, collecting, and monitoring personally identifiable information: From the dark web to the surface web. In *Proceedings of the 2020 IEEE International Conference on Intelligence and Security Informatics*. IEEE, 1–6.

[20] Steve Morgan. 2017. *2017 Cybercrime Report*. Technical Report. Cybersecurity Ventures. Retrieved 11 Feb., 2022 from http://www.cybersecurityventures.com/hackerpocalypse-cybercrime-report-2016.

[21] Vinod Nair and Geoffrey E. Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning*. 807–814.

[22] Zahra Nouri and Mahdi Rezaei. 2020. Deep-CAPTCHA: A deep learning based CAPTCHA solver for vulnerability assessment. *Available at SSRN 3633354* (2020). Accessed 11 Feb., 2022.

[23] Fabio Pierazzi, Ghita Mezzour, Qian Han, Michele Colajanni, and V. S. Subrahmanian. 2020. A data-driven characterization of modern android spyware. *ACM Transactions on Management Information Systems* 11, 1, Article 4 (2020), 38 pages. DOI:https://doi.org/10.1145/3382158

[24] Brian M. Powell, Gaurav Goswami, Mayank Vatsa, Richa Singh, and Afzel Noore. 2014. fgCAPTCHA: Genetically optimized face image CAPTCHA 5. *IEEE Access* 2 (2014), 473–484. https://ieeexplore.ieee.org/document/6807630.

[25] Awang Hendrianto Pratomo, Anggit Ferdita Nugraha, Joko Siswantoro, and Mohammad Faidzul Nasruddin. 2019. Algorithm border tracing vs scanline in blob detection for robot soccer vision system. *International Journal of Advances in Soft Computing & Its Applications* 11, 3 (2019), 40–56.

[26] John Robertson, Ahmad Diab, Ericsson Marin, Eric Nunes, Vivin Paliath, Jana Shakarian, and Paulo Shakarian. 2017. *Darkweb Cyber Threat Intelligence Mining*. Cambridge University Press.

[27] Sagar Samtani, Ryan Chinn, Hsinchun Chen, and Jay F. Nunamaker Jr. 2017. Exploring emerging hacker assets and key hackers for proactive cyber threat intelligence. *Journal of Management Information Systems* 34, 4 (2017), 1023–1053.

[28] Anna Sapienza, Sindhu Kiranmai Ernala, Alessandro Bessi, Kristina Lerman, and Emilio Ferrara. 2018. Discover: Mining online chatter for emerging cyber threats. In *Companion Proceedings of the The Web Conference 2018*. 983–990.

[29] Rajalingappaa Shanmugamani. 2018. *Deep Learning for Computer Vision: Expert Techniques to Train Advanced Neural Networks using TensorFlow and Keras*. Packt Publishing Ltd.

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

[31] Claire Tsosie. [n.d.]. Discover Launches Social Security Number Alert Features. Retrieved from https://www.nerdwallet.com/article/credit-cards/discover-feature-alerts-social-security-number-risky-sites-dark-web. (2020/07/21).

[32] Moshe Unger, Alexander Tuzhilin, and Amit Livne. 2020. Context-aware recommendations based on deep learning frameworks. *ACM Transactions on Management Information Systems* 11, 2, Article 8 (2020), 15 pages. DOI:https://doi.org/10.1145/3386243

[33] Bo Wen, Paul Jen-Hwa Hu, Mohammadreza Ebrahimi, and Hsinchun Chen. 2021. Key factors affecting user adoption of open-access data repositories in intelligence and security informatics: An affordance perspective. *ACM Transactions on Management Information System* 13, 1 (2021), 1–24.

[34] Haiqin Weng, Binbin Zhao, Shouling Ji, Jianhai Chen, Ting Wang, Qinming He, and Raheem Beyah. 2019. Towards understanding the security of modern image captchas and underground captcha-solving services. *Big Data Mining and Analytics* 2, 2 (2019), 118–144.

[35] Xing Wu, Shuji Dai, Yike Guo, and Hamido Fujita. 2019. A machine learning attack against variable-length chinese character CAPTCHAs. *Applied Intelligence* 49, 4 (2019), 1548–1565.

[36] Madhuri Yadav and Alok Kumar. 2018. Feature extraction techniques for hand written character recognition. *International Journal of Advanced Research in Computer Science* 9, 2 (2018), 521.

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

[38] Yang Zhang, Haichang Gao, Ge Pei, Sainan Luo, Guoqin Chang, and Nuo Cheng. 2019. A survey of research on CAPTCHA designing and breaking techniques. In *Proceedings of the 2019 18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*. IEEE, 75–84.

Received November 2020; revised August 2021; accepted December 2021