AdMarks: Image Steganography Based on Adversarial Perturbation

Ye Ding(✉), Mingyu Shao(✉), Jie Wang, and Qi Wan

School of Computer Science and Technology, Dongguan University of Technology, Dongguan 523808, Guangdong, China
{dingye,2112115004,2111915024,2112015032}@dgut.edu.cn

Abstract. Steganography methods (a.k.a. *watermarks*) embed specific information into multimedia carriers for the protection of copyrights. However, conventional methods such as Least Significant Bit (LSB) and frequency domain transformation are vulnerable to tampering attacks. Thus, the resulting watermarks cannot be easily traced after release to the public. We address this problem by exploiting adversarial perturbations in a positive way. The proposed *AdMarks* system embeds adversarial perturbations generated on object detection models and encodes the "detection errors" as watermarks. Our evaluations on real-life datasets show that the proposed watermarks are barely visible by human eyes and robust under image transformations such as cropping and JPEG compression. In particular, AdMarks can generate unlimited digital watermarks on single image that ensures the uniqueness and traceability on each copy of the original media.

Keywords: Steganography · Watermark · Adversarial examples

1 Introduction

Copyright protection of digital contents is a significant research area in recent years. There are numerous ways to protect the copyright of digital content, and the watermark is one of the primary methods. Digital watermarking aims to embed specific information into the original media (or strictly, *signal*), including image, audio, and video. A general copy of the signal will also copy the embedded information. Watermarks could be visible or hidden, depending on the purpose of the watermarks. If the watermarks are intended to show copyright information, visible watermarks are often applied. On the contrary, if the authors prohibit the sharing of the original work, a hidden watermark could be applied for source tracking. For the latter form of watermark, *steganography* is an effective approach to hide information in the original image. The information could be embedded into the pixel, quantization, or spread-spectrum domain. A strong steganography algorithm should have the following desired properties: **1) Imperceptibility**, i.e., the digital watermarks should be barely visible and the watermarked image should be of high visual quality; **2) high robustness**, i.e., the embedding should remain recognizable or at least partially recognizable under tampering attacks; and **3) high capacity**, i.e., the size of embedded information should be sufficiently large. The watermarks must be identifiable by the source-generating model, but cannot be recovered by any other model, which guarantees the highly desired property of traceability. Generally speaking, *higher* capacity tends to reduce robustness and increase perceptibility of digital watermarks (Fig. 1).

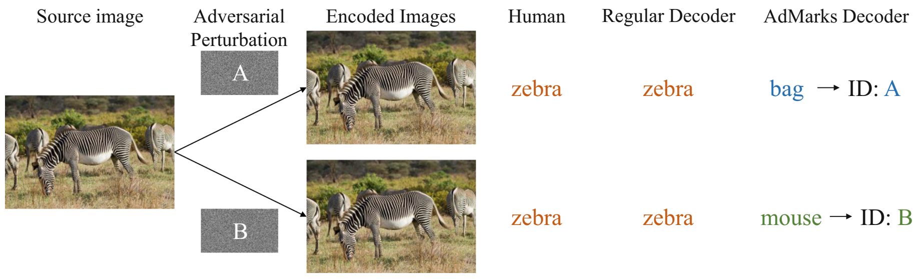


Fig. 1. AdMarks embeds different adversarial perturbations to images and encodes them as steganographic signals into different watermark IDs.

Existing steganographic watermarking methods often *embed* readable information into the source image. Despite of the embedding domains, the embedded information of steganographic watermarks should be recognizable through a *decoding* process. Meanwhile, such readable information could also draw the attention of hackers to construct targeted attacks by decoding the source information or tampering the embedded watermarks.

Inspired by recent work of adversarial examples, we propose a novel steganography method that generates the embedding signal first and then encodes it into a readable copy ID. This is fundamentally different from the conventional process where the steganography signal is typically encoded *from* the readable ID. Our main idea is to exploit an auxiliary object detector to generate adversarial perturbations as the steganography signal for embedding. Note that our method is a novel use of adversarial perturbations in wider applications other than conventional attacks and defences, and can be easily extended to other forms of source media.

Our primary contributions are: 1) We propose a novel use of adversarial perturbations for good in steganography. Specifically, we generate adversarial perturbations on an auxiliary detector as steganography signals. As the specific detection information is *not* released to the public, AdMarks examples have higher security as the watermarks can only be detected and decoded by the specific detector that generates the perturbations for embedding. 2) We design a novel encoding and decoding scheme, called e-NMI, to ensure traceable watermarks of which the uniqueness and identifiability are formally provided. 3) Our experimental results on real datasets demonstrate that AdMarks significantly improves the steganography performance in terms of the three desired properties. In particular, the resulting watermarks by AdMarks are identifiable under various tampering attacks.

2 Related Work

2.1 Digital Watermark

Digital watermark is a method that embeds specific information into the original digital signal. The embedded information should be retrievable, and the original signal should be kept readable. In general, traditional digital watermarks are embedded into two domain categories: 1) *spatial domain*, and 2) *transform domain*.

The basic way to embed digital watermarks into the spatial domain is Least Significant Bit (LSB) [10]. The main principle of the LSB is to change the significant parts of the image by slightly changing pixel intensity. For example, divide an image into pieces of the same size and embed specific watermarks into these pieces. This embedding method directly alters the pixel value of the image, which is simple with a low computation cost. However, LSB is not robust and lacks imperceptibility. In transform domain such as Discrete Cosine Transform (DCT), Discrete Wavelet Transform (DWT), and Discrete Fourier Transform (DFT), for most of these algorithms, the original image is first segmented with Expectation Maximization (EM) and then divided into pixel blocks (usually in zigzag form). The transformation of each pixel is subsequently calculated, after the transformation, a pseudorandom sequence of real numbers is embedded into the corresponding DCT/DWT/DFT domain of each segment.

More advanced algorithms include 1) Full Counter-propagation Neural Network (FCNN), which employs perceptual and visual features to enhance the transformation and embeds the highest modifiable threshold values of DCT coefficients; 2) 2-Level DWT, DCT and QR Decomposition based on optimal blocks selection, where the watermark embedding process is performed on specific blocks of the host image according to its entropy values; and 3) DNN-based watermarking schemes [1,7,16], which use an automatic encoder-based network architecture to perform the process of information embed and extraction.

2.2 Adversarial Examples

Szegedy *et al.* [13] are the first to reveal the problem of *adversarial examples*. They discovered that deep learning networks could easily misclassify an image by adding a barely perceptible perturbation. Then in their later work of studying the nature of deep networks [5], they put forward a simple and universal method to find adversarial examples called Fast Gradient Sign Method (FGSM). More follow up works include PGD [8], EAD [3], AB-FGSM [15] and ZeroGrad [4]. These works expose the security issues among the widely-used deep learning models.

To overcome the above security issues and make the deep learning models more robust, many defences against adversarial examples have been proposed later on. For example, defensive distillation [11] and adversarial training [9] are two effective methods of defense against FGSM, but are vulnerable to iterative attacks [8]. Inspired by cryptologic terms, researches are divided into two groups: textitattacks (creation of adversarial examples) and textitdefends (model protection). The robustness of both groups develops spirally. For example, Metzen *et al.* [9] proposed a "detector" network, which explicitly detects adversarial examples. However, Kurakin *et al.* [6] suggested a threat model that could create adversarial objects by printing digital adversarial examples on paper, which ultimately misleads the detector; Moreover, Athalye *et al.* [2] enhanced the above work by creating adversarial objects that are robust to varying viewing angles.

In this paper, we successfully insert adversarial attacks into the training process which ensure the results of the classification go wrong as expected. Through a carefully designed encoding process, these expected errors constitute the steganographic watermark.

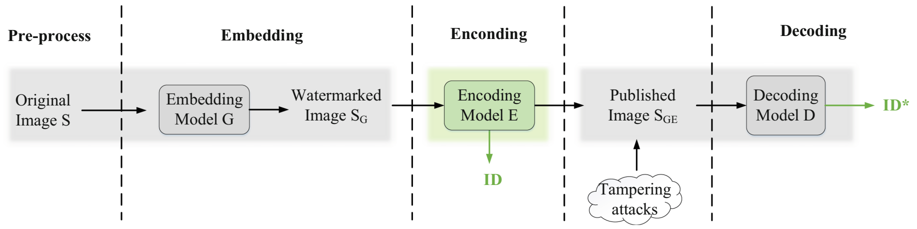


Fig. 2. The watermarking process of AdMarks. Given an original image S, the embedding model G first generates a watermark consists of adversarial perturbations, and then embeds the watermark into S_G. The encoding model E encodes an ID with respect to S_G, resulting S_{GE}. When the encoded image S_{GE} is leaked to the public, the decoding function D could identify the ID from S_{GE} through the embedded corresponding watermark.

3 AdMarks

The watermarking process of AdMarks is shown in Fig. 2. Given an original image S, the embedding model G first generates a watermark consists of adversarial perturbations, and then embeds the watermark into S, resulting S_G. Then, the encoding function E encodes an ID with respect to S_G, resulting S_{GE}. When the encoded image S_{GE} is leaked to the public and found by the system, the decoding function D could decode the ID from S_{GE} by identifying the corresponding watermark which is embedded inside S_{GE}. The watermarking process above is different from traditional watermarking process, where encoding is not necessary. Traditional methods directly embed watermarks into the source image, and the watermarks are readable as introduced in Sect. 1, decoder can directly extract the watermarks. However, in AdMarks, the object detection model cannot generate "detection errors" (i.e., the copy ID) before actually detects the objects from watermarked image S_G. Hence, AdMarks is fundamentally different from traditional watermarking methods.

In the following of this section, we will introduce the embedding, encoding and decoding details of AdMarks. Formally,

Definition 1 (Embedding). *Given an image S and a set of watermark candidates $W = \{w_1, w_2, \dots\}$, an embedding model G finds S_G such that:*

$$S_G = \underset{G(S,W^*)}{\arg\min}\ \varphi(S, G(S, W^*)) \tag{1}$$

and:

$$\begin{aligned} &W^* \subseteq W \\ &G(S, W_i^*) \neq G(S, W_j^*), \forall W_i^*, W_j^* \subseteq W, W_i^* \neq W_j^* \end{aligned} \tag{2}$$

In this paper, the size of W (denoted as $|W|$) is usually very large, hence it is not feasible to enumerate all the candidates. Equation (1) is introduced to ensure that W^* is barely visible (i.e., *imperceptibility* as mentioned in Sect. 1 on S, and φ is the *distance* (i.e., visual similarity) between two images. Equation (2) is introduced to ensure that each embedded image is unique and identifiable.

Definition 2 (Encoding). *Given a watermarked image S_G and a set of unique IDs $K = \{\varkappa_1, \varkappa_2, \dots\}$, an encoding function E finds S_{GE} such that:*

$$S_{GE} = E(S_G, \varkappa) \tag{3}$$

and:

$$E(S_G, \varkappa_i) \neq E(S_G, \varkappa_j), \forall \varkappa_i, \varkappa_j \in K, \varkappa_i \neq \varkappa_j \tag{4}$$

Similarly as Definition 1, Eq. (4) is introduced to ensure that each encoded image is unique and identifiable.

Please note that $\varphi(S, G(S, W^*))$ is usually proportional to $|W^*|$ in Definition 1, but it is not feasible to minimize $|W^*|$ because Eq. (4) will not be satisfied given a large $|K|$ and a very small $|W^*|$, i.e., it is not possible to encode a high capacity with very low perceptibility, as explained in Sect. 1.

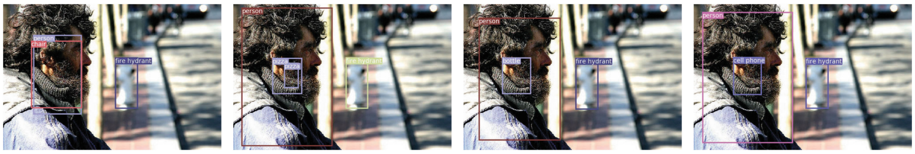
Definition 3 (Decoding). *Given an encoded image S_{GE}, a decoding function D finds the associated $\varkappa$ for S_{GE} such that:*

$$\varkappa = D(S_{GE}) \tag{5}$$

and $\varkappa$ satisfies Eq. (3) in Definition 2.

An effective watermark system must ensure that each watermarked image is imperceptible at the embedding stage, identifiable and highly capable at the encoding stage, and robust at the decoding stage. In the following of this paper, we will introduce the embedding, encoding and decoding details of AdMarks in Sects. 3.1, 3.2, and 3.3, respectively. As introduced in Sect. 1, AdMarks is designed for images, but it is extendable based on the idea of adversarial examples. In the following sections, we will use the term "image" instead of "signal" for better understanding.

3.1 Embedding



(a) `89vqng12ogn` (b) `5tf9h581alb` (c) `86ncv10r6u2` (d) `8548jm1le70`

Fig. 3. Traceable AdMarks examples with unique watermark ID's by specifying different adversarial labels for object detection in the same source image.

As shown in Fig. 2, the embedding model produces various adversarial perturbations as watermarks W, which will be embedded into the source image S and then outputs the embedded images S_G. More specifically, an *object detection model* is first introduced to locate candidate objects in the source image S, and then AdMarks specifies which object area will be attacked. Then, the *embedding model* G will generate adversarial perturbations so that the chosen object is misclassified as a target label y_t rather than its actual label y. The adversarial training of this neural network will recursively modify the perturbation w of the chosen object area until the object is misclassified with maximum confidence, and the distance $\varphi(S, G(S, W))$ between source image S and embedded image $G(S, W)$ is minimized.

For the object detection, there is no need to specify which model to be applied, since the detection process will be interfered by adversarial perturbations anyway. In this paper, we use YOLOv3 [12] as the object detector, which is a single-stage object detection algorithm with the advantage of fast detection speed. The loss of YOLOv3 is set to:

$$L_y(S_G, y_t) = \lambda_1 L_r + \lambda_2 L_o + \lambda_3 L_c \tag{6}$$

where L_r is the object coordinate information loss, including width loss and height loss, L_o is the object category confidence loss, and L_c is the classification loss.

For the embedding model, given an image S, it finds an imperceptible perturbation w and embeds it into S to obtain the perturbed image $G(S, w)$, which will mislead the object detector d to recognize the ground-true label y of the object as the target label y_t. An efficient method that generates the perturbation is

Fast Gradient Sign Method (FGSM) [5]. FGSM utilizes the signs of gradients from the classifier as the perturbation. Formally,

$$S_G = S + \epsilon \,\mathrm{sign}(\nabla_S f(\theta, S, y)) \tag{7}$$

where ϵ is the hyper-parameter used to adjust the perturbation scale, ∇_S is the derivative function with respect to S, and θ is the model parameter.

However, the perturbation generated by FGSM seriously affects the fidelity of the original image. Moreover, FGSM is an algorithm implemented on the classifier, which results in low attack success rate (i.e., the classifier f cannot be mislead). Since our goal is to achieve a high attack rate on the object detector, we introduce a novel approach by extending BIM/ILLC [6] on FGSM, called *FGSM-II*. FGSM-II repeatedly generates S_G by:

$$S_G^{n+1} = \underset{0,255}{\mathrm{clip}}\,(S + w_{n+1}) \tag{8}$$

where clip ensures that the pixel value after modification is limited to $[0, 255]$, and:

$$\begin{aligned} w_{n+1} &= \underset{-\epsilon,\epsilon}{\mathrm{clip}}(\eta \,\mathrm{sign}(\delta_{n+1})) \\ \delta_{n+1} &= \top_k(\nabla_{\mathrm{EOT}(S_G^n)} d(\mathrm{EOT}(S_G^n), y_t)) \end{aligned} \tag{9}$$

where δ_{n+1} is the top k among the retained values, d is the object detector, and clip limits the watermark size. The initial value $S_G^0 = S$. EOT is *Expectation Over Transformation* [2], indicates that certain simulated transformation attacks are performed before the actual attack. The simulated attacks mainly include a series of common spatial domain operations such as rotation, shearing, scaling, and some frequency domain noises such as Gaussian blur and JPEG compression to improve the robustness. The total loss of AdMarks is:

$$L_e = L_y(S_G, y_t) + \mathrm{mse}(S, S_G) + L_1(S, S_G) \tag{10}$$

where $\mathrm{mse}(S, S_G)$ is the mean square error of S and S_G, and $L_1(S, S_G)$ refers to the distance between S and S_G as L_1-norm. The cross entropy of mean square error and L_1-norm ensures fidelity and fool ratio.

3.2 Encoding

The purpose of encoding is to uniquely distinguish each watermarked image S_G after it is released to the public. Therefore, a natural encoding method is to find the difference between each S_G. According to FGSM-II, for any two watermarked images S_G^i and S_G^j, $\varphi(S, S_G^i) \neq \varphi(S, S_G^j)$. Thus, we can use the difference between S and S_G to distinguish and assign the ID $\varkappa$ to each S_G.

To measure the difference between S and S_G, we propose an improved NMI encoding scheme called *e-NMI (Encoded Normalized Mutual Information)*. Given the coordinate information I of source image S and I_G of embedded image S_G, the mutual information of I and I_G are calculated through:

$$\mathrm{MI}(I, I_G) = \sum_{i \in I^*} \sum_{j \in I_G^*} \Pr(i,j) \log(\frac{\Pr(i,j)}{\Pr(i)\Pr(j)}) \tag{11}$$

where I^* and I_G^* are the distinct sets of I and I_G, $\Pr(i,j)$ is the joint probability mass function of I and I_G, and $\Pr(i)$ and $\Pr(j)$ are the marginal probability mass functions of I and I_G. Now we can calculate e-NMI based on MI through:

$$\mathrm{e-NMI}(I, I_G) = \frac{1}{n} \sum_{i=0}^{n} \frac{2\,\mathrm{MI}_k(I, I_G)}{H(I^*) + H(I_G^*)} \times 10^c \tag{12}$$

where $H(I^*)$ and $H(I_G^*)$ are the cross entropy of I^* and I_G^*, MI_k is the mutual information with k retained digits after the decimal point (e.g., $0.1234 \rightarrow 123$ when $k = 3$), n is the recursion parameter, c is the length (a.k.a. *capacity*) of ID $\varkappa$. According to our experiments, larger c requires larger k and n to solve Eq. (12) and increases the calculation time.

For the coordinate information I, considering the characteristics of adversarial perturbations, AdMarks encodes the following perturbation information: 1) digitized label of each object; 2) normalized bounding box coordinate information of each object.

Finally, we pick $\varkappa$ through $\varkappa = \mathrm{e-NMI}(I, I_G)$ and uniquely assign $\varkappa$ to S_G via encoding model E, resulting $S_{GE} = E(S_G, \varkappa)$. In production environment, in order to reduce the storage space of $\varkappa$, we represent $\varkappa$ as a base-36 string, e.g., 147036 as `35gc`. The encoded examples are shown in Fig. 3.

3.3 Decoding

Unlike previous works, it is not necessary for AdMarks to determine the size of embedded watermarks in advance, thus no watermark extraction is required. According to Sect. 3.2, each encoded image S_{GE} is associated with an unique ID $\varkappa$. Since AdMarks does not modify the contents of embedded image S_G in the encoding process, encode the encoded image S_{GE} again will result in the same ID $\varkappa$, i.e., $E(S_G, \varkappa) = E(S_{GE}, \varkappa)$. Hence in this paper,

$$\varkappa = D(S_{GE}) = \mathrm{e-NMI}(I, I_{GE}) \tag{13}$$

where I_{GE} is the coordinate information of the encoded image S_{GE}.

Once $\varkappa$ is extracted from the target S_{GE}, AdMarks will find the corresponding S_G from the database containing all mappings of $\varkappa$ and S_G. It is unnecessary for $\varkappa$ to exactly match one record in the database, and any string similarity calculation methods can be applied while finding the corresponding S_G.

4 Evaluation

Table 1. Imperceptibility measured by PSNR (first row) and SSIM (second row) between the original and watermarked images. Higher value is better.

Method	L_2-norm $(0, 50]$	L_2-norm $(50, 150]$	L_2-norm $(150, 200]$	Avg.
RWS	35.6	34.7	33.7	33.5
[10]	0.9327	0.9289	0.9206	0.9254
StegaStamps	36.4	32.6	29.1	31.0
[14]	0.9773	0.8955	0.7365	0.9071
HiDDeN	42.4	41.1	38.7	41.3
[16]	0.9790	0.9773	0.9626	0.9768
HiDDeN*	34.2	32.4	31.0	32.0
[16]	0.9630	0.9528	0.9488	0.9502
AdMarks	**50.1**	**49.0**	**48.9**	**49.6**
	0.9967	**0.9953**	**0.9947**	**0.9959**

In this paper, we use *PSNR* and *SSIM* as the evaluation metrics of imperceptibility, and *DA (Detect Accuracy)* as the evaluation metric of robustness. DA verifies whether the corresponding watermarks can be successfully tracked under various distortion attacks. It is calculated by comparing the PSNR value of the leaked watermarked image and the watermarked image in the database, and find the one with the highest PSNR value is the corresponding watermarked image.

For comparison, we choose one representative traditional watermarking algorithm RWS [10] and two DNN-based watermarking algorithms: 1) HiDDeN [16] (without image distortion) and HiDDeN* [16] (with combination of standard image distortions), and 2) StegaStamps [14]. The watermark size of the above algorithms is set to 32-bit.

All models are trained and evaluated on the MS COCO dataset and all images are resized to 256×256. As introduced in the encoding and decoding sections, we use e-NMI as the encoding and decoding functions of AdMarks. In addition, we set capacity $c = 16$ (indicates actual capacity 10^{16}), recursion $n = 3$, and retained digits $k = (2, 4, 6)$ for each recursion. Unless otherwise stated, we set $\eta = 1$ and $\epsilon = 1$ for perturbation length, $k = 1$ for top-k retained gradients, $n = 40$ for watermark generating iterations.

Imperceptibility. AdMarks does not embed information into the entire original image, but selects a target area in the image and generates a unique watermark in the target area. Moreover, AdMarks does not require noise training and utilizes less perturbations to generate watermarks, thus intuitively more imperceptible. In this paper, we evaluate imperceptibility through the perturbation size in terms of L_2-norm between the original and watermarked images. We use 4,000 random images from the MS COCO dataset for comparison, and the results are shown in Fig. 4. It is clear that the perturbations generated by AdMarks are much smaller than StegaStamps and HiDDeN*, and thus leads to a higher PSNR and SSIM as show in Table 1.

Robustness. The robustness test is performed on more than 4,000 random images with a perturbation length of the L_2-norm between $(50, 150)$. Here only

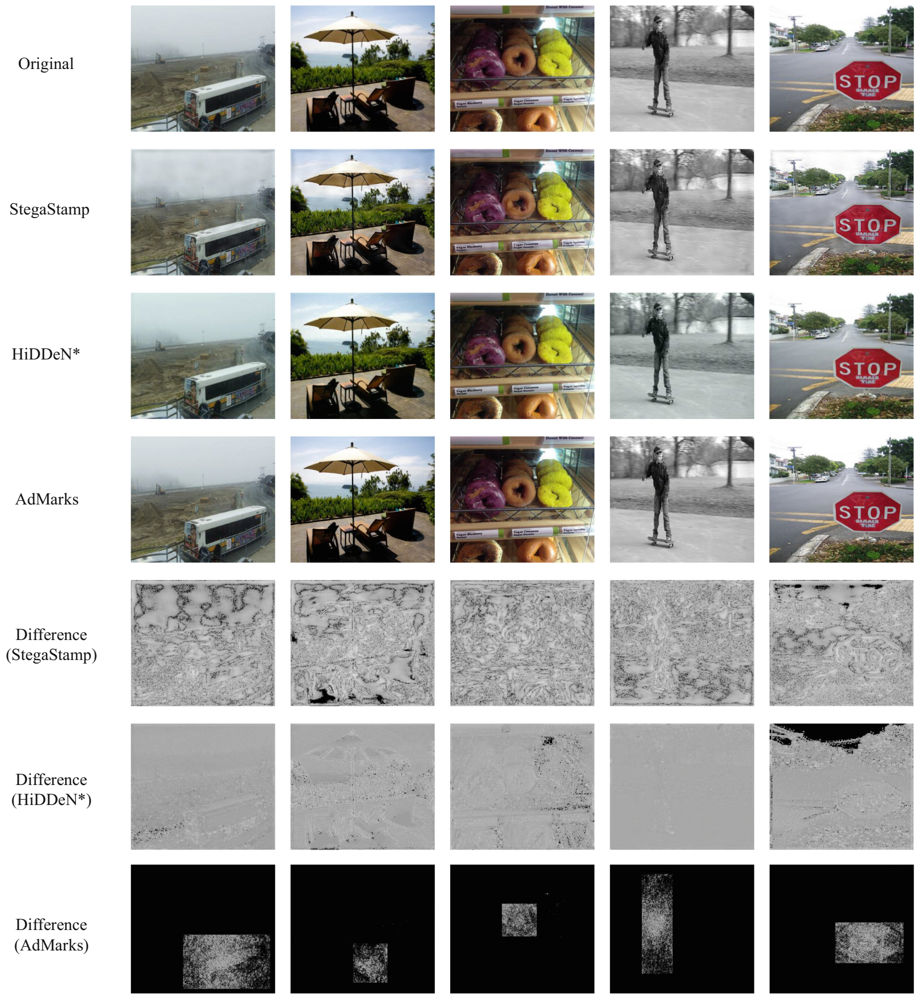


Fig. 4. Source images and their watermarks generated by three comparing steganography methods. The normalized differences between the original image and the watermarked images are also shown at the bottom. The gray level is between 0 and 1 where black indicates no change.The proposed AdMarks requires least modification at the pixel level. (Color figure online)

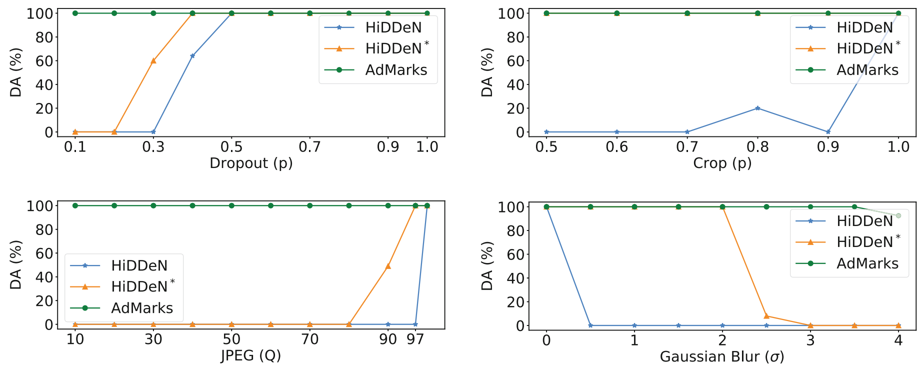


Fig. 5. Evaluation of robustness in terms of detection accuracy under different intensity distortions.

HiDDeN with PSNR value above 40db and its variant HiDDeN* is considered as the baseline. It can be seen in Fig. 5 that without noise training, AdMarks performs significantly better than HiDDeN across a different range of distortion intensities. However, AdMarks reaches comparable performance on crop, outperforms HiDDeN* on dropout, JPEG and Gaussian blur. Under the different intensity of a variety of distortions, AdMarks shows excellent performance. Especially for JPEG compression, with a $Q = 10$ quality factor, AdMarks still reaches 100% detection accuracy. Even under tampering attacks of Gaussian blur with $\sigma = 4$, the detection accuracy of AdMarks remains 92.5% while that of HiDDeN and HiDDeN* drops quickly to 0.

Capacity. In this paper, capacity is the ability to generate a copy ID from the set of encoded candidates. As introduced in Sect. 3.2, capacity c can be adjusted according to Eq. (12), and a larger c will take more time to solve Eq. (12). Since the size of target labels $|Y_t|$ can be almost unlimited if the training dataset is large enough, the resulting adversarial perturbations can be considered as unlimited according to FGSM-II as shown in the embedding section. Thus, the size of candidate IDs $|K|$ can also be unlimited, and it gives a very large capacity to AdMarks.

5 Conclusion

We propose AdMarks to generate imperceptible and traceable watermarks by encoding adversarial perturbations in a configurable area of a source image using object detection models. We demonstrate that AdMarks has superior imperceptibility compared to the state-of-the-art, requiring smaller signal strength for embedding in the majority of test samples. It maintains high detection accuracy under different tampering attacks with varying strengths. AdMarks also has unlimited capacity due to the adversarial perturbation space for generating steganography signals. Notably, it can also be applied to images without identifiable objects, as long as perturbations can be generated and imposed on specific subject areas.

Acknowledgments. This work was supported by National Key Research and Development Program of China (No. 2022YFF0606303).

References

1. Ahmadi, M., Norouzi, A., Karimi, N., Samavi, S., Emami, A.: Redmark: framework for residual diffusion watermarking based on deep networks. Expert Syst. Appl. **146**, 113157 (2020)
2. Athalye, A., Engstrom, L., Ilyas, A., Kwok, K.: Synthesizing robust adversarial examples. In: 35th International Conference on Machine Learning, pp. 284–293. PMLR (2018)
3. Chen, P., Sharma, Y., Zhang, H., Yi, J., Hsieh, C.: EAD: elastic-net attacks to deep neural networks via adversarial examples. In: AAAI Conference on Artificial Intelligence, pp. 10–17. AAAI Press (2018)
4. Golgooni, Z., Saberi, M., Eskandar, M., Rohban, M.H.: Zerograd: costless conscious remedies for catastrophic overfitting in the FGSM adversarial training. Intell. Syst. Appl. **19**, 200258 (2023)
5. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. In: Bengio, Y., LeCun, Y. (eds.) 3rd International Conference on Learning Representations (2015)
6. Kurakin, A., Goodfellow, I.J., Bengio, S.: Adversarial examples in the physical world. In: 5th International Conference on Learning Representations (2017)
7. Luo, X., Zhan, R., Chang, H., Yang, F., Milanfar, P.: Distortion agnostic deep watermarking. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 13545–13554. IEEE (2020)
8. Madry, A., Makelov, A., Schmidt, L., Tsipras, D., Vladu, A.: Towards deep learning models resistant to adversarial attacks. In: 6th International Conference on Learning Representations (2018)
9. Metzen, J.H., Genewein, T., Fischer, V., Bischoff, B.: On detecting adversarial perturbations. In: 5th International Conference on Learning Representations (2017)
10. Nasir, I., Weng, Y., Jiang, J.: A new robust watermarking scheme for color image in spatial domain. In: Yétongnon, K., Chbeir, R., Dipanda, A. (eds.) Third International IEEE Conference on Signal-Image Technologies and Internet-Based System, pp. 942–947. IEEE Computer Society (2007)
11. Papernot, N., McDaniel, P.D., Wu, X., Jha, S., Swami, A.: Distillation as a defense to adversarial perturbations against deep neural networks. In: IEEE Symposium on Security and Privacy, pp. 582–597. IEEE Computer Society (2016)
12. Redmon, J., Farhadi, A.: Yolov3: an incremental improvement. CoRR abs/1804.02767 (2018). http://arxiv.org/abs/1804.02767
13. Szegedy, C., et al.: Intriguing properties of neural networks. In: 2nd International Conference on Learning Representations (2014)
14. Tancik, M., Mildenhall, B., Ng, R.: Stegastamp: invisible hyperlinks in physical photographs. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 2114–2123. IEEE (2020)

15. Wang, Y., Liu, J., Chang, X., Wang, J., Rodríguez, R.J.: AB-FGSM: adabelief optimizer and fgsm-based approach to generate adversarial examples. J. Inf. Secur. Appl. **68**, 103227 (2022)
16. Zhu, J., Kaplan, R., Johnson, J., Fei-Fei, L.: HiDDeN: hiding data with deep networks. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11219, pp. 682–697. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01267-0_40