Watermarking Neural Networks with Watermarked Images

Hanzhou Wu, *Member, IEEE*, Gen Liu, Yuwei Yao and Xinpeng Zhang

***Abstract*—Watermarking neural networks is a quite important means to protect the intellectual property (IP) of neural networks. In this paper, we introduce a novel digital watermarking framework suitable for deep neural networks that output images as the results, in which any image outputted from a watermarked neural network must contain a certain watermark. Here, the host neural network to be protected and a watermark-extraction network are trained together, so that, by optimizing a combined loss function, the trained neural network can accomplish the original task while embedding a watermark into the outputted images. This work is totally different from previous schemes carrying a watermark by network weights or classification labels of the trigger set. By detecting watermarks in the outputted images, this technique can be adopted to identify the ownership of the host network and find whether an image is generated from a certain neural network or not. We demonstrate that this technique is effective and robust on a variety of image processing tasks, including image colorization, super-resolution, image editing, semantic segmentation and so on.**

***Index Terms*—Watermarking, neural networks, deep learning, image transformation, information hiding.**

I. Introduction

RECENT advances in deep learning (DL) [1] have led to great success in a variety of fields, e.g., visual computing [2]–[4], speech recognition [5]–[8], and natural language processing [9], [10]. Major enterprises such as Microsoft, Apple, and Google, have already deployed DL models in their commercial products to provide higher-quality and smart services. However, when we are experiencing the many advantages of profound changes brought by DL, new problems and chances are arising as well, among which securing DL models is one of the most important yet quite challenging topic. Considering such a real application scenario, an enterprise has developed a DL based product and distributes it to customers for profits. According to the agreement, the customers have the right to use the service by themselves, but cannot provide the product and service to others for commercial use without permission from the enterprise. Obviously, how to protect the intellectual property (IP) of the product is a critical problem.

As a means to information hiding [11], digital watermarking [12] enables us to hide secret information into a digital object without impairing the use of the object. It has extensive applications, e.g., integrity authentication [13], copyright protection [14], source tracking [15] and tracing traitors [16] (yet another common way for tracing traitors is fingerprinting [17] such as Tardos codes [18], [19]). Therefore, a straightforward idea to protect the IP of DL models is applying digital watermarking. Traditional watermarking algorithms are often limited to media objects, e.g., digital image is still the most popular cover type for watermarking because of its wide distribution over social networks [20], [21]. As designing a watermarking system takes into account the cover characteristics, traditional media based watermarking algorithms cannot be directly applied to the DL models. Though some works [22]–[24] have combined DL into a media watermarking system, they actually aim to protect the media content, rather than the used DL model. It has motivated us to investigate the IP protection of DL models.

Actually, there are increasing watermarking works designed for protecting DL models in recent years. They can be roughly categorized into two categories, i.e., carrying watermarks by network weights and by classification labels of the trigger set.

Carrying watermarks by network weights. Embedding a watermark into the weights of a DL model is a straightforward idea, which has been investigated in the existing works. E.g., Uchida *et al.* [25] first propose to mark deep neural networks (DNNs) with a parameter regularizer allowing the watermark to be embedded during model training phase. Wang *et al.* [26] further improve Uchida *et al.*'s work by adding an independent neural network to project network weights to watermark space. However, it cannot avoid the ambiguity attack. To this end, Fan *et al.* [27] propose novel passport-based DNN ownership verification schemes, which are relatively robust to network modifications and resilient to the ambiguity attack. Rouhani *et al.* [28] propose DeepSigns, which, unlike previous works, is an end-to-end IP protection framework that enables developers to systematically insert digital watermarks in the pertinent DL model before distributing the model. Zhang *et al.* [29] propose a watermark implanting approach to infuse watermarks into DL models, and design a remote verification mechanism to identify ownership, enabling the DL models to learn specially crafted watermarks at training and activate with pre-specified predictions when observing watermark patterns at inference.

Carrying watermarks by classification labels of trigger set. This approach aims to insert a backdoor into the network to be protected, and only the owner can activate the watermark extraction procedure. For example, Adi *et al.* [30] introduce a backdoor based watermarking framework by incorporating the author's signature in the process of training DNNs. The resulting watermarked DNN behaves in a different, predefined pattern when given any signed inputs, accordingly proving the authorship. Shafieinejad *et al.* [31] also present an approach for watermarking DNNs in the way of backdoor, which works for

H. Wu is with the School of Communication and Information Engineering, Shanghai University, Shanghai 200444, China. (Email: h.wu.phd@ieee.org)

G. Liu is with the School of Communication and Information Engineering, Shanghai University, Shanghai 200444, China. (Email: 704105800@qq.com)

Y. Yao is with the School of Communication and Information Engineering, Shanghai University, Shanghai 200444, China (Email: happyyuwei1994@qq.com)

X. Zhang is with the School of Communication and Information Engineering, Shanghai University, Shanghai 200444, China; and also with the School of Computer Science, Fudan University, Shanghai 200433, China. (Corresponding author, Email: xzhang@shu.edu.cn)

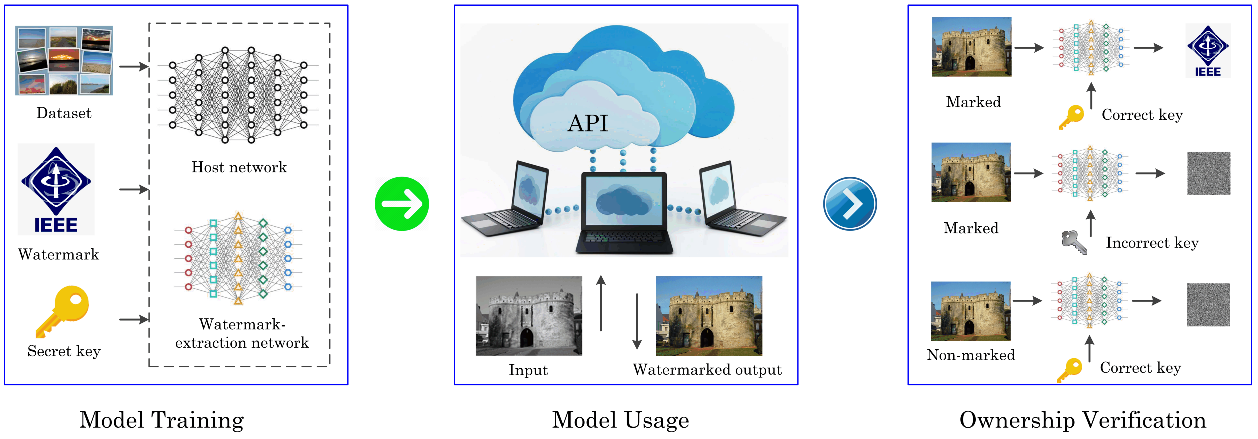

Fig. 1. The application scenario of applying the proposed method. The entire process can be summarized in three phases: model training, model usage, and ownership verification. Any image outputted from the network trained by our framework must carry a watermark. By detecting watermarks in the outputted images, it can be used to identify the ownership of networks and find whether an image is generated from a certain network or not.

general classification tasks and can be combined with current learning algorithms, demonstrating that embedded watermarks are relatively robust to removal attacks such as fine-tuning and pruning. One could refer to [32]–[34] for more related works.

There has no doubt that, the above-mentioned watermarking algorithms have moved the IP protection of DL models ahead rapidly. However, in the real-world, a company or person may steal a DL model and then only provides the API service to users for profits without permission. In this case, the owner cannot identify the ownership if the watermark was previously embedded into the DL model by directly modifying network weights since he/she cannot access the network weights and structure. One may use traditional media watermarking methods or adversarial network [35] to process the output for model protection [36]. However, as the operation does not rely on the internal network details, once the network itself was stolen, it cannot protect the model. Though the classification labels of trigger set can alleviate it, the watermark capacity is relatively too low to carry a sufficient payload. Moreover, they are often limited to classification tasks, and cannot be applied directly to popular generative tasks such as image transformation.

In this work, we propose a novel watermarking framework to protect DNNs, which is applicable for networks that produce images as results, such as image colorization and editing. Any image outputted from a DNN trained by our framework must contain a certain watermark. In order to extract the watermark, we design a watermark-extraction network, where the loss is based on mean accuracy of the watermark. The DNN to be protected (called *host network*) and the watermark-extraction network are trained together. During the training phase, the parameters of the host network will be updated according to its own loss and the watermark loss. Meanwhile, the parameters of the watermark-extraction network will be updated according to only the watermark loss. After training the two networks, only the host network will be released and the watermark-extraction network will be kept as a secret. Experiments have shown that this technique is effective and robust on a wide variety of image tasks, including image colorization, super-resolution, image editing and semantic segmentation.

The contributions of this paper are summarized as follows:

- We present a novel watermarking technique to DNNs that produce images as outputs. It can identify the ownership of DNNs and identify whether an image is generated from a certain network. Moreover, when a DNN model is not allowed to feed trigger images (e.g., standard GANs [35] and VAEs [37] have only a low dimensional latent vector as input), the proposed work can still embed a watermark since the proposed work does not rely on the input of the host DNN, which is a significant advantage compared to trigger set based methods (that require trigger inputs).
- The proposed work can resist common attacks including image cropping and noise adding, which has been verified by experimental results. Even though the image was tampered with a relatively high degree, the hidden watermark can be still sufficiently extracted. It is due to the reason that tampered samples were used for adversarial training.
- We introduce a secret key for watermark extraction. The secret key is only kept by the IP holder. The watermark-extraction network works when the correct key is provided. Once the watermark-extraction network is stolen, the watermark is still undetectable without the key.

The rest structure of this paper is organized as follows. We detail the proposed method in Section II, followed by extensive experiments and analysis in Section III. Finally, we conclude this paper and provide further discussion in Section IV.

II. Proposed Method

In this section, we propose a novel watermarking framework suitable for DNNs that output images as results, in which any image outputted from this network must carry a certain watermark. Fig. 1 shows an application scenario of embedding a watermark in a DNN model by our framework. The entire process can be summarized in three phases: model training, model usage, and ownership verification.

Model Training. During the training phase, the host network G and the watermark-extraction network E are trained together by optimizing a combined loss. The host network G

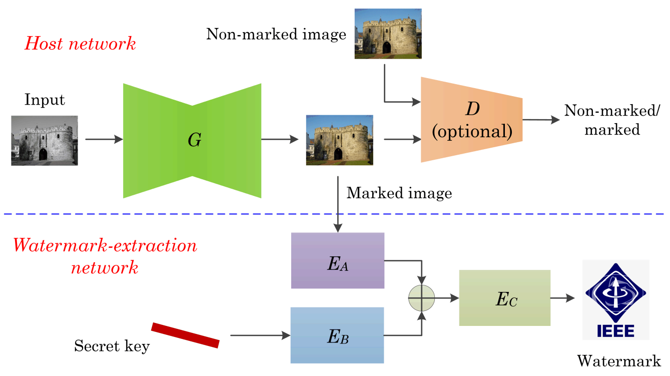

Fig. 2. Sketch for the proposed watermarking framework.

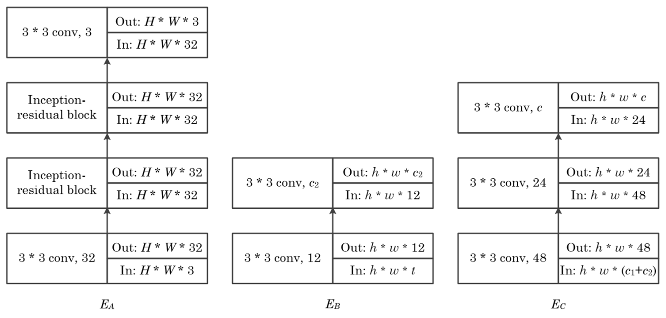

Fig. 3. The detailed structural information for E_A, E_B, and E_C.

learns to accomplish the original task while embedding a watermark into the outputted images. The watermark-extraction network E learns to extract the hidden watermark from the output of the host network. It is noted that, in order to train these two networks simultaneously, an image dataset and a secret watermark w_{raw} (together with a key k) are used. The details of G and E will be given in the subsequent subsections.

Model Usage. After the two networks mentioned above are trained, the *marked* host network can be released to accomplish a particular image task. And, the watermark-extraction network will be kept secretly. It is pointed that, unlike other ordinary networks, for the marked host network, it will output an image containing a watermark, which can be used for ownership verification, source tracking, and model identification. The watermark can be either visible or invisible.

Ownership Verification. In case of copyright disputes, the IP holder can extract a watermark w_{ext} from the "*marked*" output of G by using E together with a secret key k for ownership verification. Mathematically, for any input $x \in X$, the output of G, denoted by $G(x)$, contains a secret watermark $w_{\text{ext}} = E(G(x), k)$ that is quite close to the expected watermark w_{raw} according to a distance measure. If one uses an incorrect key $k' \neq k$ or a non-marked image x', the distance (distortion) between the extracted watermark and the original watermark is expected to be significantly large.

A. Structural Design

As shown in Fig. 2, the host network consists of a generator G and a discriminator D (*optional*). The watermark-extraction network consists of three sub-networks, allowing the hidden watermark to be reconstructed from the output of G. We show details about the structures of all networks below.

1) Host network: The generator G could be arbitrary structure related to image transformation. Mathematically, given two image domains X and Y, G aims to learn a mapping, i.e., $G : X \rightarrow Y$, which can be trained with an image dataset including lots of diverse image-pairs by supervised learning. With a trained G, an image can be generated, which is visually close to the ground truth (if it exists). Moreover, optionally, a discriminator D can be utilized for optimizing G. Since we focus on the generality of the proposed framework, we will not discuss a particular network structure to be protected. A variety of structures are supported by the proposed framework, including ResNet [4], U-Net [3], CGAN [38], and so on.

2) Watermark-extraction network: The extraction network E accepts an image and a secret key k as input. In the structure of E, we combine three convolutional neural networks (CNNs) E_A, E_B, and E_C to learn the extraction function. The procedure of the forward propagation follows the following settings:

- E_A accepts a digital image sized $H \times W \times 3$ as the input, and outputs an image with the identical size. The output image would be resized to $h \times w \times c_1$. Here, we have $c_1 = \lfloor \frac{3HW}{hw} \rfloor$, and we consider the watermark as an image sized $h \times w \times c$, e.g., we have $c = 3$ for color images.
- The secret key k is a binary string sized l_k, which can be determined from a seed or specified rule. In the proposed work, the secret key is randomly generated according to a secret seed such that it can be flatten as a tensor sized $h \times w \times t$, where $l_k = h \times w$ and $t \geq 1$. The tensor can be divided to disjoint blocks and the elements in each block have the same value since CNNs have superior ability to learn block based features [1], [4]. A larger block size can facilitate model training but reduce the significant size of the key. From a trade-off point, we use $4 \times 4 \times t$ (tunable) as the block size and $t = 1$. Therefore, before feeding k into E_B, it is resized to a tensor sized $h \times w \times t$. Thereafter, E_B will output a feature map sized $h \times w \times c_2$, where c_2 is empirically set to be 48 in our experiments.
- The outputs of E_A and E_B will be further concatenated to constitute a feature map sized $h \times w \times (c_1 + c_2)$, which will be fed into E_C. The output of E_C represents the reconstructed watermark, whose size is $h \times w \times c$.

Fig. 3 shows the structural information of E_A, E_B and E_C. In detail, E_A starts with a convolutional layer, followed by two inception-residual blocks [39], and finally connected by a convolutional layer. As shown in Fig. 4, in order to partition the patterns in the generated image into different channels, the inception-residual block is utilized, which consists of 1×1, 3×3, 5×5 convolutions, and a residual connection that sums up a feature map and the input itself, so that various perception fields are included in the feature extraction. It is straightforward to understand the structures of E_B and E_C. It is noted that, ReLU [40] is used here for all layers except for the output (which uses 'tanh') as the activation function since ReLU does not activate all neurons at the same time and thus has shown superior performance in accelerating convergence

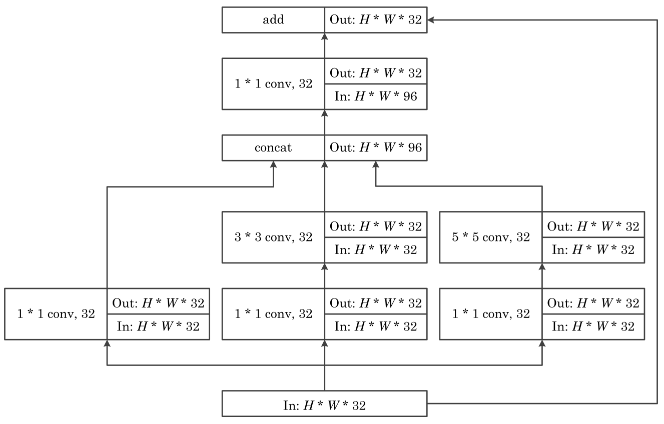

Fig. 4. The detailed structural information for the inception-residual block.

[41]. Moreover, using the 'tanh' for the output can prevent the DNN from modeling large values [42]. However, it is still open for us to design other effective activation functions.

B. Loss Function

The host network and the watermark-extraction loss will be trained according to the task loss $\mathcal{L}_t$ and watermark loss $\mathcal{L}_w$.

Watermark Loss. From the model owner's point of view, all images can be divided into two groups: one is generated by the host network, the other one is not generated by the host network. We denote the former as S_1 and the latter as S_2. Each image in S_1 contains a watermark, which can be extracted by E together with the correct secret key. Each image in S_2 reveals nothing about the watermark, meaning that, the output of E by feeding any image in S_2 is expected to be noise-like.

Therefore, since we hope that a particular watermark can be always extracted from any image in S_1, the distance between the extracted watermark and the target watermark should be as small as possible. Namely, we expect to minimize:

$$\mathcal{L}_w^{(1)} = \frac{1}{|S_1|} \sum_{G(x)\in S_1} ||E(G(x), k) - w||_p^p, \tag{1}$$

where w denotes the target watermark, and k means the secret key. On the other hand, we hope that only random noises can be extracted from the images in S_2, implying that, the distance between an extracted watermark and a random-noise image is small. It indicates that, we expect to minimize:

$$\mathcal{L}_w^{(2)} = \frac{1}{|S_2|} \sum_{x_i\in S_2} ||E(x_i, k) - w_z||_p^p, \tag{2}$$

where w_z reveals nothing about w and is randomly generated in advance. In our experiments, w_z is an all-zero matrix for simplicity. Furthermore, the correct key k can be used to ensure the security of the watermark extraction procedure. It means that, when the used key is incorrect, the watermark extraction procedure will fail. In this way, we hope to minimize:

$$\mathcal{L}_w^{(3)} = \frac{1}{|S_1|} \sum_{G(x)\in S_1, k_x\neq k} ||E(G(x), k_x) - w_z||_p^p, \tag{3}$$

Accordingly, the watermark loss here is finally designed as:

$$\mathcal{L}_w = \alpha\mathcal{L}_w^{(1)} + \beta\mathcal{L}_w^{(2)} + \gamma\mathcal{L}_w^{(3)}, \tag{4}$$

TABLE I
TASKS, ARCHITECTURES, DATASETS AND THEIR SUBSET SPLITS.

Task	DNN	Dataset	Training/validation/testing
Paint Transfer	[54]	[49]	180,000/10,000/10,000
Image Editing	[50]	[50]	380/100/100
Super-resolution	[43]	[51]	118,287/20,000/20,679
Colorization	[38]	[52]	7,000/500/689
Semantic Segmentation	[38]	[53]	2,975/1,000/1,025

where α, β and γ are tunable parameters so that different tasks can be properly balanced [45], [46]. It is required to set $\alpha > \beta$ and $\alpha > \gamma$ since the primary task of the watermark-extraction network is to reconstruct the watermark. Therefore, we use $\alpha = 1$ and $\beta = \gamma = 0.5$ in default. We use L_1 distance for the watermark loss. One may also use L_2 distance.

Task Loss. The primary goal of the host network here to be protected is to accomplish a specific task. We below present several loss functions that are popular in different kinds of tasks and will be used in our experiments. The task loss can be one of them or their combinations, depending on the task.

- *Pixel loss [38]:* It is defined as the distance between the output image and the target image by using L_1 distance, rather than L_2, so as to encourage less blurring.
- *Discriminator loss:* The loss function can not only be mathematical formula, but also network, called discriminator D, as shown in Fig. 2. The discriminator loss [35] is calculated by subtracting the fake loss from the real loss. The generator is basically trying to trick the discriminator, while the discriminator learns to classify between fake and real. In this paper, "fake" means "marked", and "real" means "non-marked (images)".
- *Perceptual loss:* Rather than encouraging the pixels of the output image to exactly match the pixels of the target image, Johnson *et al.* [43] instead encourage them to have similar feature representations as computed by the loss network, such as a well-trained VGG-16 network [44].

Thus, to train the host network and the watermark-extraction network simultaneously, the total loss can be expressed as:

$$\mathcal{L} = \mathcal{L}_t + \theta\mathcal{L}_w, \tag{5}$$

where θ is a tunable parameter, and $\mathcal{L}_t$ depends on the task. In default, we use $\theta = 1$ since the original task and the watermark extraction task are equally important in this paper.

Remark: For the original task, on the one hand, designing a universal host network structure suited to arbitrary image task is a very challenging problem that has not been well addressed. On the other hand, altering a particular host network structure also requires us to design the matched loss function(s), which should take into account the original task and is not the focus of proposed work. Therefore, we do not limit the host network to any particular structure, which shows better generalization ability. It is possible that, the original task was also controlled by using a secret key or a trigger pattern. For example, one may use a trigger pattern [47] to control the host network to perform its original task. Additionally, though our experiments have verified the superiority and applicability of the proposed

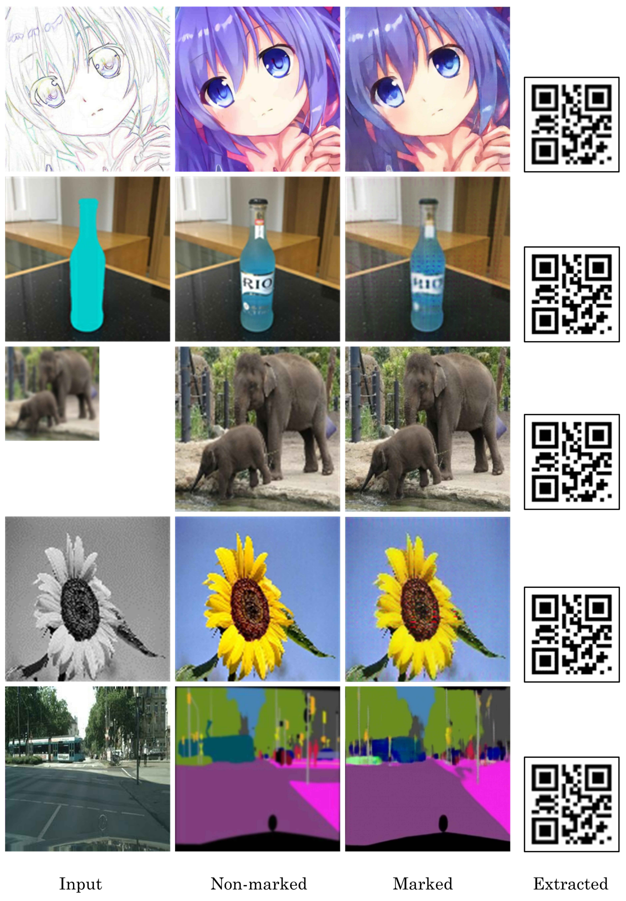

Fig. 5. Examples for embedding a binary QR code for the five tasks: paint transfer, image editing, super resolution (×4), colorization, and semantic segmentation. First column: input, second column: non-marked images (generated by neural networks), third column: marked images, fourth column: extracted watermarks (ignoring the black lines on borders and the relative ratios between the watermarks and the marked images). The SSIMs between the non-marked images and the corresponding marked images are determined as 0.896, 0.886, 0.965, 0.975 and 0.809, respectively (from the first row to the fifth row).

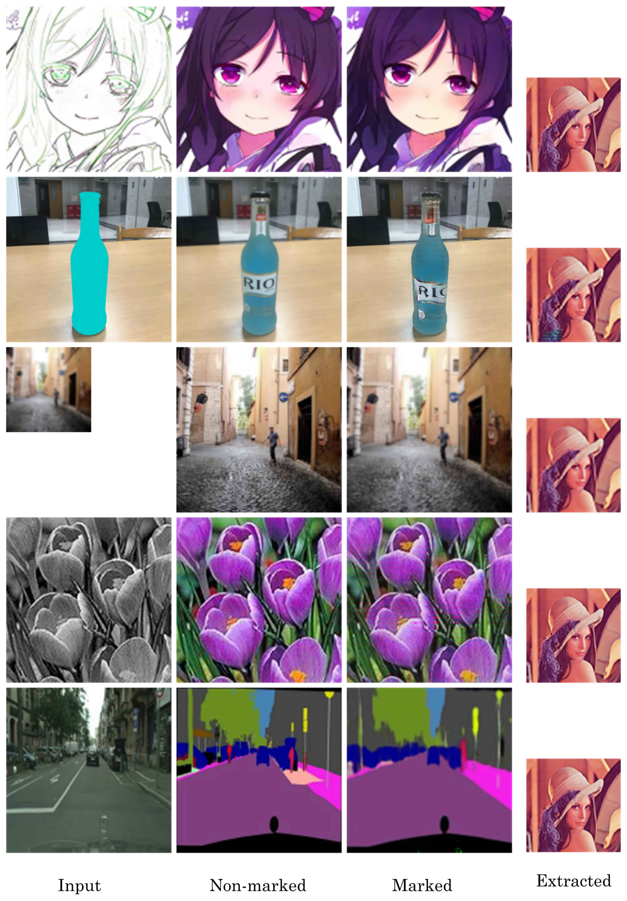

Fig. 6. Examples for embedding a color image for the five tasks: paint transfer, image editing, super resolution (×4), colorization, and semantic segmentation. First column: input, second column: non-marked images (generated by neural networks), third column: marked images, fourth column: extracted watermarks (ignoring the relative ratios between the watermarks and the marked images). The SSIMs between the non-marked images and the corresponding marked images are determined as 0.898, 0.917, 0.945, 0.955 and 0.718, respectively (from the first row to the fifth row).

watermark-extraction structure, it should be admitted that it is still open for us to design other efficient network structures.

III. Experimental Results and Analysis

In this section, we present experimental results for performance evaluation and analysis.

A. Datasets and Setup

To explore the generality of our framework, we implement our evaluations on a variety of tasks and datasets, including both graphic tasks such as image editing, and vision tasks such as semantic segmentation. Table 1 shows the detailed information for the datasets used in this paper and their split information for experiments. We note that our framework is suitable for different sizes of the training sets, e.g., the paint transfer task is trained on Danbooru2019 [49] consisting of more than 100 thousand images, while the image editing task is trained on RIO [50] consisting of less than 400 images. Both perform well after embedding a certain watermark.

The used DNN architectures are also provided in Table 1. Both inputs and outputs of the DNNs follow the same size, i.e., $256 \times 256 \times 3$. The height and width of the watermark to be embedded are all fixed as 64, if no otherwise specified.

As shown in Table 1, For paint transfer, we follow the structure in sketch transfer [54]. For image editing, we follow the structure in product placement [50]. For super-resolution, we use the perceptual loss [43]. For colorization and segmentation, we use Pix2Pix [38]. Note that, in our experiments, we regard the segmentation task as the generative task, and only L_1 loss is used as discussed in Pix2Pix [38].

During model training, the host network and the watermark-extraction network are trained together. The ADAM optimizer [55] is adopted for training, where the learning rate is set as 1.0×10^{-8}, $L_1 = 0.5$, $L_2 = 0.999$. Our implementation uses TensorFlow and cuDNN, trained on a single Titan RTX GPU. In Eq. (3), the incorrect key k_x is randomly generated for each sample, and the number of different bits between k_x and the correct key k is controlled to be a random integer no more than a small threshold T_d in each time, e.g., we empirically use $T_d = 8$ in our experiments. The advantage is that, using a small T_d forces the watermark-extraction network to learn to capture the slight difference between two different keys. Thus,

TABLE II
PERFORMANCE OF NETWORK FIDELITY AND WATERMARK RELIABILITY FOR EMBEDDING BINARY WATERMARKS. BOTH PSNR_0 (DB) AND SSIM_0 ARE DETERMINED BETWEEN THE IMAGES GENERATED BY NON-MARKED NEURAL NETWORKS AND THE CORRESPONDING GROUND TRUTHS. BOTH PSNR_1 (DB) AND SSIM_1 ARE DETERMINED BETWEEN THE IMAGES GENERATED BY MARKED NEURAL NETWORKS AND THE CORRESPONDING GROUND TRUTHS. BOTH PSNR_2 (DB) AND SSIM_2 ARE DETERMINED BETWEEN THE IMAGES GENERATED BY MARKED NEURAL NETWORKS AND THAT GENERATED BY NON-MARKED NEURAL NETWORKS. BER IS DETERMINED AS THE PERCENTAGE OF ERRORS BETWEEN THE EXTRACTED WATERMARK AND THE CORRESPONDING GROUND TRUTH. ALL EXPERIMENTAL RESULTS SHOWN IN THIS TABLE ARE MEAN VALUES.

Task	BER	PSNR_0	PSNR_1	PSNR_2	SSIM_0	SSIM_1	SSIM_2
Paint Transfer	0.0083	22.62	20.13	21.26	0.642	0.618	0.814
Image Editing	0.0000	33.56	31.68	36.63	0.923	0.901	0.981
Super-resolution	0.0092	21.35	20.42	31.69	0.637	0.613	0.934
Colorization	0.0000	30.27	29.77	28.54	0.814	0.807	0.952
Semantic Segmentation	0.0075	23.30	21.56	24.55	0.819	0.810	0.876

TABLE III
PERFORMANCE OF NETWORK FIDELITY AND WATERMARK RELIABILITY FOR EMBEDDING COLOR WATERMARKS. BOTH PSNR_0 (DB) AND SSIM_0 ARE DETERMINED BETWEEN THE IMAGES GENERATED BY NON-MARKED NEURAL NETWORKS AND THE CORRESPONDING GROUND TRUTHS. BOTH PSNR_1 (DB) AND SSIM_1 ARE DETERMINED BETWEEN THE IMAGES GENERATED BY MARKED NEURAL NETWORKS AND THE CORRESPONDING GROUND TRUTHS. BOTH PSNR_2 (DB) AND SSIM_2 ARE DETERMINED BETWEEN THE IMAGES GENERATED BY MARKED NEURAL NETWORKS AND THAT GENERATED BY NON-MARKED NEURAL NETWORKS. PSNR (DB) IS DETERMINED BETWEEN THE EXTRACTED WATERMARK AND THE CORRESPONDING GROUND TRUTH. ALL EXPERIMENTAL RESULTS SHOWN IN THIS TABLE ARE MEAN VALUES.

Task	PSNR	PSNR_0	PSNR_1	PSNR_2	SSIM_0	SSIM_1	SSIM_2
Paint Transfer	34.22	22.62	20.18	20.51	0.642	0.626	0.792
Image Editing	38.54	33.56	31.46	34.75	0.923	0.905	0.968
Super-resolution	31.71	21.35	20.03	28.81	0.637	0.610	0.909
Colorization	32.35	30.27	29.04	26.95	0.814	0.797	0.922
Semantic Segmentation	31.21	23.30	21.72	23.36	0.819	0.735	0.830

TABLE IV
PERFORMANCE OF NETWORK FIDELITY AND WATERMARK RELIABILITY FOR EMBEDDING BINARY WATERMARKS WITH DIFFERENT SIZES. PSNR (DB) IS DETERMINED BETWEEN THE OUTPUTTED MARKED IMAGES AND THE GROUND TRUTHS. BER_w IS DETERMINED AS THE PERCENTAGE OF ERRORS BETWEEN THE EXTRACTED WATERMARKS AND THE GROUND TRUTHS. ALL EXPERIMENTAL RESULTS SHOWN IN THIS TABLE ARE MEAN VALUES.

Task	32^2		64^2		96^2		128^2		256^2	
	PSNR	BER_w	PSNR	BER_w	PSNR	BER_w	PSNR	BER_w	PSNR	BER_w
Paint Transfer	22.34	0.0035	20.13	0.0083	21.42	0.0112	20.94	0.0150	21.44	0.0224
Image Editing	33.97	0.0000	31.68	0.0000	33.43	0.0026	35.52	0.0072	31.57	0.0083
Super-resolution	23.28	0.0040	20.42	0.0092	21.88	0.0183	23.15	0.0197	22.07	0.0232
Colorization	29.70	0.0000	29.77	0.0000	30.13	0.0060	29.71	0.0071	29.85	0.0087
Semantic Segmentation	24.66	0.0000	21.56	0.0075	21.98	0.0080	25.52	0.0126	24.15	0.0181

TABLE V
PERFORMANCE OF NETWORK FIDELITY AND WATERMARK RELIABILITY FOR EMBEDDING COLOR WATERMARKS WITH DIFFERENT SIZES. PSNR (DB) IS DETERMINED BETWEEN THE OUTPUTTED MARKED IMAGES AND THE GROUND TRUTHS. PSNR_w (DB) IS DETERMINED BETWEEN THE EXTRACTED WATERMARKS AND THE GROUND TRUTHS. ALL EXPERIMENTAL RESULTS SHOWN IN THIS TABLE ARE MEAN VALUES.

Task	32^2		64^2		96^2		128^2		256^2	
	PSNR	PSNR_w	PSNR	PSNR_w	PSNR	PSNR_w	PSNR	PSNR_w	PSNR	PSNR_w
Paint Transfer	22.90	34.50	20.18	34.22	22.84	32.39	23.05	29.45	22.12	27.72
Image Editing	31.10	38.59	31.46	38.54	32.22	34.89	31.28	25.79	30.25	23.84
Super-resolution	22.49	33.82	20.03	31.71	20.94	29.25	20.90	26.38	20.54	25.22
Colorization	30.29	38.35	29.04	32.35	28.42	27.30	32.18	24.14	32.06	20.08
Semantic Segmentation	24.06	32.79	21.72	31.21	24.12	29.27	23.14	27.91	22.82	23.53

the watermark-extraction network will be sensitive to the key.

B. *Qualitative Evaluation*

In this section, the basic results of two implementations by the proposed framework will be introduced. One is embedding a binary image, and the other one is embedding a color image. Both watermarks are invisible, which should be detected by the watermark-extraction network. A binary image is a digital image that has only two possible values for each pixel. A color image includes RGB components for each pixel.

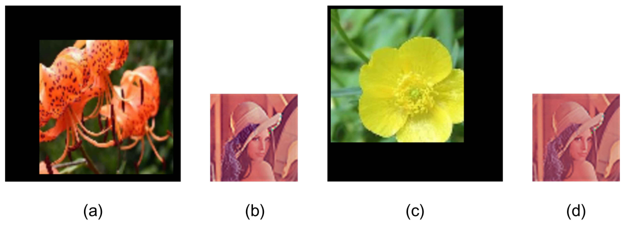

Fig. 7. Examples for cropping the marked images: (a, c) the cropping images, (b, d) the extracted watermarks from corresponding cropped images.

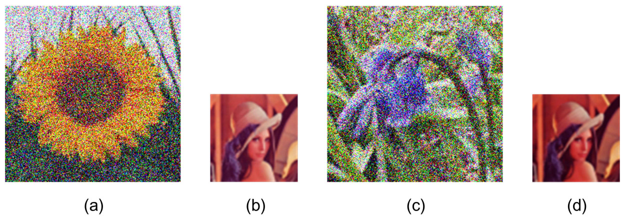

Fig. 8. Examples for inserting noise to the marked images: (a, c) the noised images, (b, d) the extracted watermarks from corresponding noised images.

Fig. 5 shows visual examples of embedding a binary image. Fig. 6 shows visual examples of embedding a color image. It can be seen from Fig. 5 and Fig. 6 that, all the non-marked networks and marked networks can successfully complete the corresponding original tasks, indicating that, the watermark-extraction networks do not suppress the original task. The non-marked images and the marked images are visually similar to each other. It implies that, the watermark-extraction networks do not significantly impair the original tasks. It is also seen that, the visual quality of all extracted watermarks are satisfactory to human visual system, meaning that, the proposed work would be effective in identifying the ownership of DNNs.

However, we can also find that different original tasks show different subjective quality of generated images. For example, for paint transfer, the non-marked images and marked images have better subjective visual quality than that for image editing though image editing can achieve the high SSIMs. Two reasons can be derived for explaining this normal phenomenon. The first one is the size of training set [56]. Namely, more training samples can lead to better performance since more training data enable the DNN to learn domain knowledge to generate images with better quality. Therefore, it can be inferred from Table 1 that, due to the small size of training set, image editing can lead to the relatively worse subjective quality of generated images compared to paint transfer. The second reason is that, a host DNN itself has a bias on the image task. Clearly, in this paper, we did not design any new host DNN for a specific image task. There may be better DNNs designed for a specific image task in the literature. This indicates that, the host DNNs tested in this paper may be not optimal for the corresponding image tasks. Therefore, for better evaluation, it is desirable to compare the difference between non-marked images and ground truths as well as the difference between marked images and ground truths, which can skip the bias caused by the host DNNs themselves. It leads us to present more results below.

C. Quantitative Evaluation

We first evaluate the proposed work in terms of network fidelity and watermark reliability. The network fidelity means the performance on the original task of the host neural network should be not significantly degraded after embedding a secret watermark. We use structural similarity-index (SSIM) [48] and peak signal-to-noise rate (PSNR) to measure the quality of the images generated by the marked neural network. Watermark reliability means the watermark should be effectively detected and extracted. It is important because the IP holder is thereby able to detect any guilty use of the model. In case of using binary images as the watermarks, we use bit error rate (BER) to evaluate the quality of the extracted watermarks. The BER is computed as the percentage of errors between the binarization of extracted watermark and the ground truth. In case of using color images, we employ PSNR to evaluate the image quality.

In experiments, we train the non-marked DNNs for baseline, and train the corresponding marked DNNs for comparison. We evaluate the fidelity of the host DNNs with PSNRs and SSIMs. The number of the generated images for a specific task can be found in Table I (i.e., the number of testing images). The mean PSNR and mean SSIM are determined as the results. We evaluate the reliability of the extracted watermarks with BERs (for binary watermarks) and PSNRs (for color watermarks). The mean BER and mean PSNR are determined as the results.

Table II and Table III have shown the experimental results. It can be observed that the BER values for binary watermarks are small and the PSNR values for color watermarks are all larger than 31 dB. It implies that the reconstructed watermarks have good visual quality, and therefore can be very effective in protecting the IP of DNN models. It can be also seen that, the overall difference between "PSNR_0" and "PSNR_1" as well as that between "SSIM_0" and "SSIM_1" are small. It indicates that, the proposed work does not significantly degrade the performance of the original task. We can also find that, the performance for "SSIM_2" significantly outperforms "SSIM_0" and "SSIM_1". And, in most cases, the performance for "PSNR_2" outperforms "PSNR_0" and "PSNR_1". It implies that, regardless of the performance impact caused by the host DNNs themselves, the proposed work can well maintain the global statistical distribution of the generated images. In other words, the proposed work does not impair the performance of the used host DNNs on the original task (though the used host DNNs may not perform well on the original task). In addition, by comparing Table II and Table III, it can be further observed that, in terms of "PSNR_2" and "SSIM_2", the performance for binary watermarks are better than the performance for color watermarks. It is due to the reason that color watermarks contain more visual information than binary watermarks and therefore require more representation ability of the host DNNs.

D. Capacity

To analyze the capacity, we retrain all the neural networks separately by embedding watermarks with different sizes. We vary the watermark size from 32×32 to 256×256. In our experiments, both binary watermarks and color watermarks are considered. We compute the mean PSNRs and mean BERs for

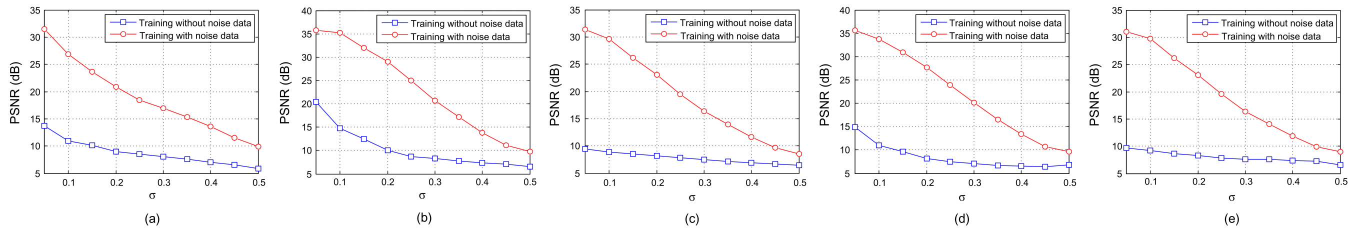

Fig. 9. The mean PSNR between the original watermark and the reconstructed watermark due to different σ for each image task: (a) paint transfer, (b) image editing, (c) super-resolution, (d) colorization and (e) semantic segmentation.

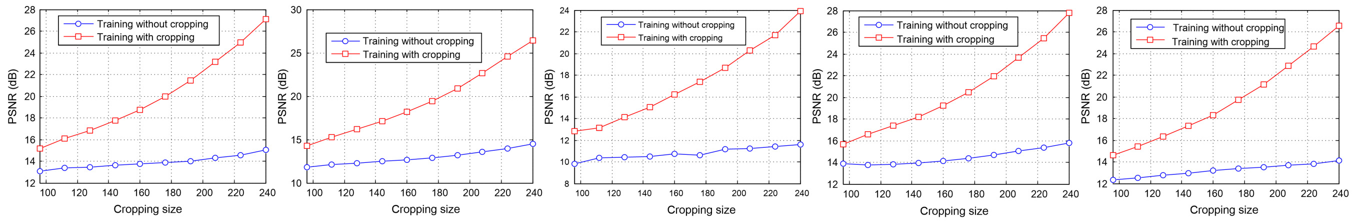

Fig. 10. The mean PSNR between the original watermark and the reconstructed watermark due to different cropping rates for each image task: (a) paint transfer, (b) image editing, (c) super-resolution, (d) colorization and (e) semantic segmentation.

the extracted watermarks, and the mean PSNRs for the marked images outputted by the host DNNs. Table IV and Table V have shown the experimental results. It could be observed from these Tables that, different image tasks result in different performance. However, for a specific task, both the network fidelity and watermark reliability can be kept in a high level when the watermark size increases. Especially, even the size of the watermark is identical to the size of the outputted image, the mean PSNRs for extracted watermarks are all above 20 dB, showing that, our work indeed achieves superior performance.

E. Robustness

Attackers may attempt to remove the embedded watermark by applying image processing operations to the marked output. We consider two common attacks, i.e., cropping and adding noise. To this end, we propose to use cropped or noised images to train the DNN to resist against the corresponding attack. For cropping attack, we randomly crop the input images for training. Namely, given an image sized $256\times256\times3$, we keep a randomly selected region sized $h_{\text{crop}}\times w_{\text{crop}}\times3$ unchanged, and set the values of all rest pixels as zeros. For simplicity, we use $h_{\text{crop}}=w_{\text{crop}}\in\{128,144,160,176,192,208,224,240\}$. We use the color image *Lena* sized $64\times64\times3$ used previously as the watermark. For evaluation, we determine the PSNR between the original watermark and the watermark reconstructed from a cropped marked image. The mean PSNR for the test images is used as the result. Fig. 7 shows two examples for cropping. For noise attack, we add Gaussian noise to each input image with $\mu=0$ and random $\sigma\in(0,0.5]$ for training. For evaluation, we determine the PSNR between the original watermark and the watermark reconstructed from a noised marked image. The mean PSNR is used as the result. Fig. 8 shows two examples for adding Gaussian noise. Fig. 9 and Fig. 10 further show the experimental results. It can be concluded that, both adding noise and cropping the inputs during training can significantly improve the robustness of retrieving the embedded watermark though the PSNRs decline when the attack degree increases during testing. It indicates that, in applications, it would be quite desirable to accordingly alter the inputs for model training to resist the specific attack.

F. System Security

In the proposed watermarking system, a secret key is used to secure the watermark reconstruction procedure. It is necessary to evaluate the sensitivity of the secret key. In theory, similar to cryptography, it is required that only the correct key can lead to the successful extraction of the embedded watermark. For any incorrect key, the corresponding reconstructed watermark should reveal nothing about the original watermark. However, unlike cryptography based on rigorous mathematical derivation and analysis, a DNN corresponds to a learning system aiming to learn domain knowledge from given training samples. The learning performance relies on the complexity of the task, the structural information of the DNN, training samples, parameter initialization and optimization. Any potential factor may cause performance degradation [1]. Therefore, from the viewpoint of practice, it is basically required that, the correct key allows the watermark to be successfully extracted and a very few errors in the key lead to the failure in watermark extraction. Fig. 11 has shown the mean PSNRs between the original watermarks and the extracted watermarks due to a different number of error bits in the secret key (whose size is set as 1024) for different tasks. It can be observed that, the PSNR values fall rapidly with the increment of error bits in the key. It can be inferred that the watermark cannot be extracted even a very few bits in the key are changed. We provide the box-plots to display the distribution of the data points shown in Fig. 11. As shown in Fig. 12, it can be observed that, the number of outliers (whose values are larger than the corresponding maximum value) is no more than 4 for all the five tasks, which has verified the aforementioned analysis. In fact, for those who cannot access the secret key, they can only guess the secret key, which will

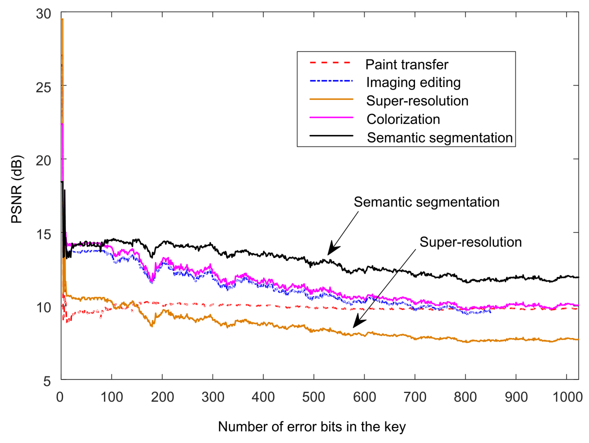

Fig. 11. The mean PSNRs between the original watermarks and the extracted watermarks due to a different number of error bits in the key for all tasks.

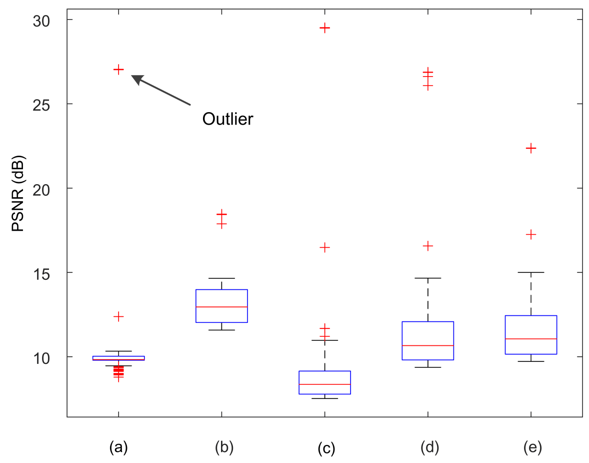

Fig. 12. The mean PSNR distribution due to a different number of error bits in the key: (a) paint transfer, (b) semantic segmentation, (c) super-resolution, (d) image editing, and (e) colorization.

result in around 50% error-bits in the secret key and therefore will surely fail in watermark extraction. Further improving the sensitivity of the secret key will be also investigated in future.

IV. Conclusion and Discussion

Model protection against IP infringement is quite important to preserve the competitive advantage of the IP holders. We propose a novel framework to protect the DNN models, which is suitable for image-outputted networks, in which any image outputted from a watermarked DNN must contain a certain watermark. This technique can be used to identify the ownership of DNNs and find whether an image is generated from a certain network. We have demonstrated that this technique is effective and robust on many image tasks. In addition, a secret key is used to control the watermark extraction, which can enhance the security of the watermarking system. Moreover, based on our experiments, it is suggested to adding additional operations such as adding noise to the generated images during training to resist real-world attacks, which could significantly improve the robustness. Digital watermarking is an effective means for protecting the ownership of digital products. It has been widely used in commercial areas and also supported by legal terms, e.g., in China, digital watermarking can be used to address the IP disputes. With the rapid development of deep learning, protecting the IP of DNN models will be very necessary. We hope this work could make contribution to the IP protection of DNN models, and inspire more advanced works in the future.

Acknowledgement

The authors would like to sincerely thank the anonymous reviewers for their quite valuable comments and suggestions for improving this paper. This work was partly supported by National Natural Science Foundation of China under Grant Nos. 61902235, U1636206, U1936214 and 61525203. It was also supported by "Chen Guang" project under Grant No. 19CG46, co-funded by Shanghai Municipal Education Commission and Shanghai Education Development Foundation.

References

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436-444, 2015.
[2] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," In: *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pp. 1-9, 2015.
[3] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: convolutional networks for biomedical image segmentation," In: *Proc. Int. Conf. Medical Image Computing and Computer-Assisted Intervention*, pp. 234-241, 2015.
[4] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," In: *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pp. 770-778, 2016.
[5] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 82-97, 2012.
[6] N. Morgan, "Deep and wide: multiple layers in automatic speech recognition," *IEEE Trans. Audio, Speech, Language Process.*, vol. 20, no. 1, pp. 7-13, 2012.
[7] O. Abdel-Hamid, A. Mohamed, H. Jiang, and G. Penn, "Applying convolutional neural networks concepts to hybrid NN-HMM model for speech recognition," In: *Proc. IEEE Int. Conf. Acoust., Speech and Signal Process.*, pp. 4277-4280, 2012.
[8] O. Abdel-Hamid, A. Mohamed, H. Jiang, L. Deng, G. Penn, and D. Yu, "Convolutional neural networks for speech recognition," *IEEE/ACM Trans. Audio, Speech, Language Process.*, vol. 22, no. 10, pp. 1533-1545, 2014.
[9] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," In: *Proc. 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 1715-1725, 2016.
[10] Y. Wu, M. Schuster, Z. Chen *et al.*, "Google's neural machine translation system: bridging the gap between human and machine translation," *arXiv Preprint arXiv:1609.08144*, 2016.
[11] F. A. P. Petitcolas, R. J. Anderson, and M. G. Kuhn, "Information hiding - a survey," *Proc. IEEE*, vol. 87, no. 7, pp. 1062-1078, 1999.
[12] M. Asikuzzaman, and M. R. Pickering, "An overview of digital video watermarking," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 9, pp. 2131-2153, 2018.
[13] H. Wu, Y. Shi, H. Wang, and L. Zhou, "Separable reversible data hiding for encrypted palette images with color partitioning and flipping verification," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 27, no. 8, pp. 1620-1631, 2017.
[14] X. Liu, C. Lin, and S. Yuan, "Blind dual watermarking for color images' authentication and copyright protection," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 5, pp. 1047-1055, 2016.
[15] H. Fang, W. Zhang, Z. Ma, H. Zhou, S. Sun, H. Cui, and N. Yu, "A camera shooting resilient watermarking scheme for underpainting documents," *IEEE Trans. Circuits Syst. Video Technol.*, online, 2019.
[16] F. Chaabane, M. Charfeddine, W. Puech, and C. B. Amaf, "A QR-code based audio watermarking technique for tracing traitors," In: *Proc. European Signal Process. Conf. (EUSIPCO'15)*, pp. 51-55, 2015.

[17] D. Milano, "Content control: digital watermarking and fingerprinting," *White Paper, Rhozet, A Business Unit of Harmonic Inc.*, 11 pages, 2012.

[18] T. Furon, and M. Desoubeaux, "Tardos codes for real," In: *Proc. IEEE Workshop Inf. Forensics Security*, pp. 24-29, 2014.

[19] T. Laarhoven, "Nearest neighbor decoding for tardos fingerprinting codes," In: *Proc. ACM Workshop Inf. Hiding Multimed. Security*, pp. 182-187, 2019.

[20] H. Wu, H. Wang, and Y. Shi, "PPE-based reversible data hiding," In: *Proc. ACM Workshop Inf. Hiding Multimed. Security*, pp. 187-188, 2016.

[21] H. Wu, H. Wang, and Y. Shi, "Dynamic content selection-and-prediction framework applied to reversible data hiding," In: *Proc. IEEE Workshop Inf. Forensics Security*, pp. 1-6, 2016.

[22] W. Sun, J. Zhou, Y. Li, M. Cheung, and J. She, "Robust high-capacity watermarking over online social network shared images," *IEEE Trans. Circuits Syst. Video Technol.*, Early Access, 2020.

[23] J. Zhu, R. Kaplan, J. Johnson, and F. Li, "HiDDeN: Hiding data with deep networks," *arXiv Preprint arXiv:1807.09937*, 2018.

[24] X. Luo, R. Zhan, H. Chang, F. Yang, and P. Milanfar, "Distortion agnostic deep watermarking," *arXiv Preprint arXiv:2001.04580*, 2020.

[25] Y. Uchida, Y. Nagai, S. Sakazawa, and S. Satoh, "Embedding watermarks into deep neural networks," In: *Proc. ACM Int. Conf. Multimed. Retrieval*, pp. 269-277, 2017.

[26] J. Wang, H. Wu, X. Zhang, and Y. Yao, "Watermarking in deep neural networks via error back-propagation," In: *IS&T Electronic Imaging, Media Watermarking, Security and Forensics*, to appear, 2020.

[27] L. Fan, K. W. Ng, and C. S. Chan, "Rethinking deep neural network ownership verification: embedding passports to defeat ambiguity attacks," In: *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, pp. 4714-4723, 2019.

[28] B. Rouhani, H. Chen, and F. Koushanfar, "DeepSigns: A generic watermarking framework for IP protection of deep learning models," *arXiv Preprint arXiv:1804.00750*, 2018.

[29] J. Zhang, Z. Gu, J. Jang, H. Wu, M. Stoecklin, H. Huang, and I. Molloy, "Protecting intellectual property of deep neural networks with watermarking," In: *Proc. Asia Conf. Comput. Commun. Security*, pp. 159-172, 2018.

[30] Y. Adi, C. Baum, M. Cisse, B. Pinkas, and J. Keshet, "Turning your weakness into a strength: watermarking deep neural networks by backdooring," *arXiv Preprint arXiv:1802.04633*, 2018.

[31] M. Shafieinejad, J. Wang, N. Lukas, X. Li, and F. Kerschbaum, "On the robustness of the backdoor-based watermarking in deep neural networks," *arXiv Preprint arXiv:1906.07745*, 2019.

[32] Z. Li, C. Hu, Y. Zhang, and S. Guo, "How to prove your model belongs to you: a blind-watermark based framework to protect intellectual property of DNN," *arXiv Preprint arXiv:1903.01743*, 2019.

[33] R. Namba, and J. Sakuma, "Robust watermarking of neural network with exponential weighting," *arXiv Preprint arXiv:1901.06151*, 2019.

[34] H. Chen, B. D. Rouhani, and F. Koushanfar, "BlackMarks: black-box multi-bit watermarking for deep neural networks," *arXiv Preprint arXiv:1904.00344*, 2019.

[35] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," In: *Proc. Int. Conf. Neur. Inf. Process. Syst.*, pp. 2672-2680, 2014.

[36] J. Zhang, D. Chen, J. Liao, H. Fang, W. Zhang, W. Zhou, H. Cui, and N. Yu, "Model watermarking for image processing networks," *arXiv Preprint arXiv:2002.11088*, 2020.

[37] D. Kingma, and M. Welling, "Auto-encoding variational bayes," *arXiv Preprint arXiv:1312.6114v10*, 2014.

[38] P. Isola, J. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial nets," In: *Proc. IEEE Conf. Comput. Vis. Patt. Recogn.*, pp. 5967-5976, 2017.

[39] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, Inception-ResNet and the impact of residual connections on learning," In: *Proc. AAAI Conf. Artificial Intell.*, pp. 4278-4284, 2017.

[40] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," In: *Proc. Int. Conf. Artificial Intell. Stat.*, pp. 315-323, 2011.

[41] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," In: *Proc. Adv. Neur. Inf. Proc. Syst.*, pp. 1097-1105, 2012.

[42] G. Xu, H. Wu, and Y. Shi, "Structural design of convolutional neural networks for steganalysis," *IEEE Signal Process. Lett.*, vol. 23, no. 5, pp. 708-712, 2016.

[43] J. Johnson, A. Alahi, and F. Li, "Perceptual losses for real-time style transfer and super-resolution," In: *Proc. European Conf. Comput. Vis.*, pp. 694-711, 2016.

[44] K. Simonyan, and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv Preprint arXiv:1409.1556*, 2014.

[45] I. Malkiel, and L. Wolf, "MTAdam: automatic balancing of multiple training loss terms," *arXiv Preprint arXiv:2006.14683*, 2020.

[46] Z. Chen, V. Badrinarayanan, C. Lee, and A. Rabinovich, "GradNorm: gradient normalization for adaptive loss balancing in deep multitask networks," In: *Proc. Int. Conf. Machine Learning*, pp. 794-803, 2018.

[47] T. Gu, B. Dolan-Gavitt, and S. Garg, "BadNets: identifying vulnerabilities in the machine learning model supply chain," *arXiv Preprint arXiv:1708.06733*, 2017.

[48] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600-612, 2004.

[49] Danbooru2019: A Large-Scale Crowdsourced and Tagged Anime Illustration Dataset, https://www.gwern.net/Danbooru2019, Available, 2020.

[50] Y. Ding, G. Teng, Y. Yao, P. An, K. Li, and X Li, "Context-aware natural integration of advertisement object," In: *Proc. IEEE Int. Conf. Image Process.*, pp. 4689-4693, 2019.

[51] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. DollarC, and L. Zitnick, "Microsoft COCO: common objects in context," In: *European Conf. Comput. Vis.*, pp. 740-755, 2014.

[52] M. E. Nilsback, and A. Zisserman, "Automated flower classification over a large number of classes," In: *Proc. Indian Conf. Comput. Vis., Graphics and Image Process.*, pp. 722-729, 2008.

[53] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," In: *Proc. IEEE Conf. Comput. Vis. Patt. Recogn.*, pp. 3213-3223, 2016.

[54] L. Zhang, Y. Ji, and X. Lin, "Style transfer for anime sketches with enhanced residual u-net and auxiliary classifier GAN," *arXiv Preprint arXiv:1706.03319*, 2017.

[55] D. P. Kingma, and J. Ba, "Adam: a method for stochastic optimization," *arXiv Preprint arXiv:1412.6980*, 2014.

[56] S. Krig, "Ground truth data, content, metrics, and analysis," In: *Computer Vision Metrics*, Apress, Berkeley, CA, pp. 283-311, 2014.

Hanzhou Wu (M'16) received B.S. and Ph.D. from Southwest Jiaotong University, Chengdu, China, in June 2011 and June 2017, respectively. From October 2014 to October 2016, he was a visiting scholar in New Jersey Institute of Technology, New Jersey, United States. He was a research staff in Institute of Automation, Chinese Academy of Sciences, Beijing, China, from July 2017 to March 2019. Currently, he is an assistant professor in Shanghai University, Shanghai, China. His research interests include information hiding, graph theory and deep learning. He has published around 20 papers in peer journals and conferences such as IEEE TDSC, IEEE TCSVT, IEEE WIFS, ACM IH&MMSec, and IS&T Electronic Imaging, Media Watermarking, Security and Forensics. He authored one chapter for a book published by Elsevier Inc.

Gen Liu received his B.S. from Shanghai University, China, in June 2019. He is currently pursuing his M.S. in Shanghai University. His research focuses on information hiding and deep learning.

Yuwei Yao received his B.S. and M.S. from Shanghai University, China, in June 2017 and June 2020, respectively. Currently, he is a Software Engineer in Morgan Stanley (Shanghai). His research interest includes privacy protection and deep learning.

Xinpeng Zhang received B.S. from Jilin University, China, in 1995, and the M.S. and Ph.D. from Shanghai University, in 2001 and 2004, respectively. Since 2004, he has been with the faculty of the School of Communication and Information Engineering, Shanghai University, where he is currently a full-time Professor. He is also with the faculty of the School of Computer Science, Fudan University. He was with The State University of New York at Binghamton as a Visiting Scholar from 2010 to 2011, and also with Konstanz University as an experienced Researcher, sponsored by the Alexander von Humboldt Foundation from 2011 to 2012. His research interests include multimedia security, image processing, and digital forensics. He has published over 200 research papers. He was an Associate Editor for IEEE Transactions on Information Forensics and Security from 2014 to 2017.