A Watermarking-Based Framework for Protecting Deep Image Classifiers Against Adversarial Attacks

Chen Sun and En-Hui Yang
Department of Electrical and Computer Engineering, University of Waterloo
Waterloo, ON, Canada
{c97sun, ehyang}@uwaterloo.ca

Abstract

Although deep learning-based models have achieved tremendous success in image-related tasks, they are known to be vulnerable to adversarial examples—inputs with imperceptible, but subtly crafted perturbation which fool the models to produce incorrect outputs. To distinguish adversarial examples from benign images, in this paper, we propose a novel watermarking-based framework for protecting deep image classifiers against adversarial attacks. The proposed framework consists of a watermark encoder, a possible adversary, and a detector followed by a deep image classifier to be protected. Specific methods of watermarking and detection are also presented. It is shown by experiment on a subset of ImageNet validation dataset that the proposed framework along with the presented methods of watermarking and detection is effective against a wide range of advanced attacks (static and adaptive), achieving a near zero (effective) false negative rate for FGSM and PGD attacks (static and adaptive) with the guaranteed zero false positive rate. In addition, for all tested deep image classifiers (ResNet50V2, MobileNetV2, InceptionV3), the impact of watermarking on classification accuracy is insignificant with, on average, 0.63% *and* 0.49% *degradation in top 1 and top 5 accuracy, respectively.*

1. Introduction

In recent years, Deep Neural Networks (DNNs) have demonstrated tremendous success for many image related tasks, such as image classification and face recognition. Unfortunately, DNNs are also known to be vulnerable to adversarial examples—subtly crafted, but imperceptible modifications of benign inputs which, once fed into DNNs, can lead DNNs to produce incorrect outputs. Specifically, given an original benign image x, a small perturbation can be easily crafted and added to x to generate a modified image x'. The output of a DNN in response to x' will be different from that of the DNN in response to x. Such x' is an adversarial example for x. The existence and easy construction of adversarial examples pose significant security risks to DNNs, especially in safety-critical applications, including face recognition and autonomous driving.

To safeguard DNNs against adversarial attacks, one approach is to build a classifier that distinguishes adversarial examples from natural images. The rationale is that although the adversarial perturbations are imperceptible to human eyes, it may be still possible to design an algorithm to detect their existence. Along this line, several detection-based methods have been proposed [18, 17, 22]. Some detection-based methods focus on finding general intrinsic properties of adversarial examples. Other detection-based methods aim to train classification networks to distinguish adversarial examples from benign images.

Although the detection-based defenses mentioned above are effective, to some extent, against some specific adversarial attacks, they in general have been proven vulnerable to more advanced adaptive adversaries, which have the full knowledge of the DNN to be secured and the given detection strategy itself. Indeed, in their recent study, Carlini *et al*. examined 10 detection-based defenses proposed in recent years, and designed adaptive adversaries to defeat them all [4]. It seems that there is no general pattern or intrinsic property shared by all adversarial perturbations. Many studies suggest [11] that adversarial examples are widely distributed in the high-dimensional image space. As such, it is desirable to explore a different defense strategy.

Inspired by semi-fragile watermark [7, 10], in this paper, we propose a radically different approach for adversarial perturbation detection. Specifically, we propose a novel watermarking-based framework for protecting deep image classifiers against adversarial attacks. The proposed framework consists of a watermark encoder, a possible adversary, and a detector followed by a deep image classifier to be secured. At the watermark encoder, an original benign image is watermarked with a secret key by embedding confidential watermark bits into selected DCT coefficients of the origi-

nal image in JPEG format. The watermarked image may then go through possible adversarial attacks. Upon receiving a watermarked and possibly attacked image, the detector accepts it as a benign image and passes it to the subsequent classifier if the embedded watermark bits can be recovered with high precision, and otherwise rejects it as an adversarial example. The embedded watermark is further required to be robust to JPEG re-compression with a predefined quality threshold. Specific methods of watermarking and detection are also presented. They can be applied to protect any image task related DNN. They are also independent of adversarial perturbation distances. For many application scenarios, from Quality Control cameras in manufacturing [19, 29] to cameras and sensors in self-driving cars, it is reasonable to assume that the original benign image can be watermarked before it is attacked. For example, watermark bits can be embedded at the time of acquisition.

Our contributions in this paper are as follows:

- We propose a novel watermarking-based framework for safeguarding image task related DNNs against adversarial attacks.
- Within our proposed framework, specific methods of watermarking and detection are presented.
- Regular adversaries such as Fast-Gradient Sign Method (FGSM) [12], Projected Gradient Descent (PGD) [16] and Carlini & Wagner (CW) [5] attacks are modified to work within our proposed framework, and are further extended to attack our watermarking-based detection strategy (i.e., adaptive white-box attacks).
- We show by experiment on a subset of ImageNet validation dataset that our proposed framework along with the presented methods of watermarking and detection is effective against a wide range of advanced attacks (static and adaptive), achieving a near zero (effective) false negative rate for FGSM and PGD attacks (static and adaptive) with the guaranteed zero false positive rate.
- It is shown that for all tested deep image classifiers (ResNet50V2 [14], MobileNetV2 [23], InceptionV3 [26]), the impact of watermarking on classification accuracy is insignificant with, on average, 0.63% and 0.49% degradation in top 1 and top 5 accuracy, respectively.

2. Background

2.1. JPEG Lossy Compression

JPEG is one of the most commonly-used formats for images. The key steps of JPEG compression are introduced below:

Color space conversion: A given image is first converted from RGB color space to YCbCr color space, where Y is the luminance (pixel brightness) channel and Cb and Cr are chrominance (pixel color) channels.

Block-wise DCT: For each channel, the image is firstly divided into non-overlapping 8×8 blocks. Then, the pixel values of the block are decomposed into 64 frequency components with discrete cosine transform (DCT), namely DCT coefficients. Typically, DCT coefficients are scanned in zigzag order.In what follows, for each $i \in \{0, 1, ..63\}$, we will use $d(i)$ to denote the DCT coefficient at the ith frequency in zigzag order. Note that a lower index represents a lower frequency.

Quantization: After Block-wise DCT, 64 DCT coefficient will be further quantized to an integer multiple of their corresponding quantization step sizes as follows:

$$D_{QF}(i) = \lfloor \frac{d(i)}{Q_{QF}(i)} \rceil, \tag{1}$$

where $D_{QF}(i)$ is the ith quantized DCT coefficient integer, $\lfloor \cdot \rceil$ denotes the rounding function which returns the nearest integer, $Q_{QF}(i)$ is the quantization step size with respect to a certain JPEG quality factor (QF, ranges from 1 to 100), and the quantized DCT coefficient is equal to $D_{QF}(i)Q_{QF}(i)$. A smaller QF corresponds to higher quantization step sizes, which means worse image quality. In theory, any quantization step size other than $Q_{QF}(i)$ can also be used in Eq. (1).

2.2. Adversarial Attack

Since the seminal work of Szegedy *et al*. [27] and Biggio *et al*. [3] firstly suggested the existence of adversarial examples, various adversarial attack methods have been proposed. Two main categories of adversarial attacks are gradient-based attacks and optimization-based attacks. For gradient-based attacks, adversaries usually construct adversarial perturbations based on the gradients of the target DNN with respect to the original image x; examples include FGSM and PGD [12, 16]. PGD takes advantage of iteratively running FGSM with smaller step size, resulting in a stronger attack at the cost of heavier computation. Both FGSM and PGD yield a near 100% success rate with sufficiently large perturbation budget. On the other hand, optimization-based methods focus on optimizing an objective function, such as minimizing the perturbation and maximizing the confidence of adversarial example. Two well-known examples are CW attack and DeepFool [29]. These attacks usually introduce smaller perturbations to the image x compared with gradient-based attacks.

In addition, adversary's knowledge is also critical to attack's performance. There are three different threat models:

- **Black-box adversary** This type of adversary has no knowledge of either the DNN to be safeguarded or the defense strategy. As such, the choice of attack is limited to a few approaches such as transfer attacks [21] or query-based attacks [2, 6].
- **Static white-box adversary** A static white-box adversary has the full knowledge of the DNN to be secured (includ-

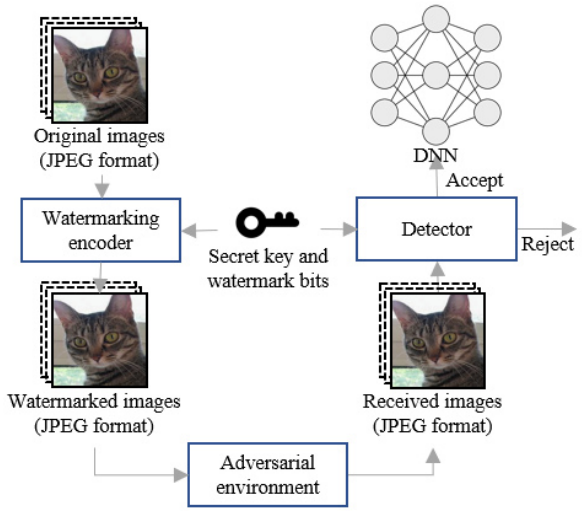

Figure 1. Illustration of the watermarking-based adversarial defense framework. Original images are first watermarked with a secret key and then possibly attacked by adversaries. Watermarked and possibly attacked images are accepted by the detector only if the embedded watermark bits are recovered with high precision.

ing its architecture and parameters), but does not know anything about the defense strategy. Adversarial examples for an image x are generated based on the DNN to be secured, the image x, a targeted output, and the allowed maximum perturbation distance from x.

- **Adaptive white-box adversary** An adaptive white-box adversary is aware of both the DNN to be secured and the defense strategy. Using this perfect knowledge, it could generate the strongest adversarial examples to defeat the defense strategy and cause the DNN to produce incorrect outputs.

3. Framework Overview

As shown in Figure 1, our proposed watermarking-based adversarial defense framework consists of a watermark encoder, a possible adversary, and a detector followed by the DNN $\mathcal{C}$ to be secured. Consider an original image x. Instead of directly exposing x to an adversarial environment, we use the watermark encoder ϕ to convert x into $x_{wm} = \phi(x, S)$, a watermarked version of x, by embedding watermark bits into x with a secret key, where S denotes both the secret key and embedded watermark bits. Note that S is kept in secret, i.e., the adversary does not have access to it. Consequently, x_{wm} may undergo adversarial attacks in the adversarial environment or some allowed legal operation before being passed to the detector. Let φ denote an adversarial attack algorithm and g denote the allowed legal operation. The image received by the detector would be either x_{wm}, $\varphi(x_{wm})$, or $g(x_{wm})$. With the help of S, the detector will further distinguish the watermarked and attacked image $\varphi(x_{wm})$ from benign images x_{wm} and $g(x_{wm})$. The received image will be fed into $\mathcal{C}$ only if it is accepted as a benign image by the detector.

As one of the most commonly-used formats for images, JPEG standard is also widely adopted in computer vision datasets and pipelines, e.g., the ImageNet dataset [9]. Since high quality JPEG compression is often acceptable or even required in practical applications, we consider JPEG re-compression with $QF \geq 50$ as the legal operation g in our framework hereafter.

There are two main objectives for our framework. First, the embedded watermark should not visibly distort the original image x, nor degrade $\mathcal{C}$'s performance significantly. Second, the images accepted by the detector should be harmless to $\mathcal{C}$. Formally, the performance metrics of the proposed framework are listed below:

- **Classification accuracy on x_{wm}** Watermarking should not significantly degrade the classification accuracy of $\mathcal{C}$. We evaluate the performance degradation of $\mathcal{C}$ by comparing the Top-1 and Top-5 accuracy on the original dataset and watermarked dataset.
- **Watermarking distortion** The watermarking distortion is evaluated using PSNR between x and x_{wm}.
- **False positive rate** If the watermarked image x_{wm} is not attacked, it should be accepted by the detector as a benign image with high probability. The false positive rate is defined as the percentage of x_{wm} that are rejected by the detector as attacked images. As we shall see later, for our proposed methods of watermarking and detection, the false positive rate is guaranteed to be 0.
- **Robustness to high quality JPEG re-compression** The embedded watermark should be robust to high quality JPEG re-compression with $QF \geq 50$, i.e., the detector should not reject $g(x_{wm})$ so as to produce a false positive case. The robustness against high quality JPEG re-compression is evaluated using the JPEG re-compression false positive rate (JRFPR), defined as the percentage of $g(x_{wm})$ that are rejected by the detector as attacked images.
- **Detection rate** The detection rate is defined as the percentage of attacked images $\varphi(x_{wm})$ that are accepted by the detector as benign images. It reflects the watermark's sensitivity to adversarial attacks.
- **Effective false negative rate** The effective false negative rate is defined as the percentage of attacked images $\varphi(x_{wm})$ that are simultaneously accepted by the detector and also successfully cause the DNN $\mathcal{C}$ to produce outputs different from those corresponding to x_{wm}. The rationale for introducing this metric is to report harmful adversarial examples only: although the adversary may bypass occasionally the detector by decreasing the perturbation budget, the strength of the resulting adversarial example

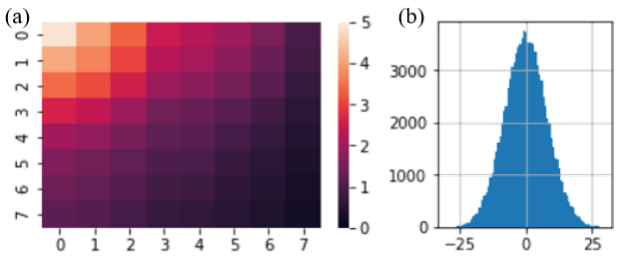

Figure 2. Coefficient-wise perturbation analysis for FGSM with $\epsilon = 8$ from the values of additive adversarial perturbations in 64 DCT coefficients of 100,000 JPEG blocks: (a) the standard derivation of perturbations per DCT coefficient; (b) distribution of perturbations at DC coefficient (DCT coefficient on the top-left corner). Note that the mean value of perturbations per DCT coefficient is more or less zero, and the perturbation energy is largely concentrated on low DCT frequencies.

will also decrease. Thus, if $\varphi(x_{wm})$ and x_{wm} share the same prediction result with respect to $\mathcal{C}$, $\varphi(x_{wm})$ is indeed harmless and should not be considered as an effective false negative case.

4. Methods of Watermarking, Detection, and Adversarial Attacks

We now describe how the watermark encoder, detector, and adversaries are designed within our proposed framework. We firstly explain our motivation of choosing watermark embedding positions (Section 4.1), and then describe the specific methods of watermarking (Section 4.2) and detection (Section 4.3). Lastly, we introduce the design of adversaries that adapt to our framework (Section 4.4).

4.1. Watermark Embedding Positions

Adversarial attacks introduce different amounts of perturbation to different frequency components of an image. As suggested in [24, 28], there are in general more perturbations in low frequency bands than in high frequency bands, at least for ImageNet models. This motivates us to embed watermark bits into DCT coefficients at low frequencies since large perturbations at low frequencies will likely destroy watermark bits embedded therein once the watermarked image is attacked.

To determine possible DCT coefficients where watermark bits can be embedded into, we performed an coefficient-wise perturbation analysis for 100,000 JPEG blocks in Luminance channel with respect to FGSM with $\epsilon = 8$. We collected the values of additive adversarial perturbation in 64 DCT coefficients of these JPEG blocks. The standard derivation of perturbations per DCT coefficient is shown in Figure 2 (a). As can be seen from Figure 2 (a), the perturbation energy introduced by FGSM is largely concentrated on low frequencies, particularly the first 16 DCT coefficients in zigzag order. Also shown in Figure 2 (b) is the distribution of perturbations at DC, which is more or less a zero-centered Gaussian distribution. Based on this analysis, we shall select the first 16 DCT coefficients in zigzag order as possible embedding positions.

4.2. Watermarking Method

To embed watermark bits into selected DCT coefficients, we invoke an invariant property of DCT coefficients from [15][Theorem 1].

Lemma 1 (DCT Invariant Property) *Let d be a DCT coefficient. If d is an integer multiple of q_0, then for any quantization step size $q < q_0$, quantizing d with q is invertible. That is, d can be fully reconstructed from its quantized value $\lfloor \frac{d}{q} \rceil q$.*

Proof Write d as $d = kq_0$, where k is an integer. It can be verified that

$$\left| d - \lfloor \frac{d}{q} \rceil q \right| \leq \frac{q}{2}. \tag{2}$$

Dividing both sides of (2) by q_0 yields

$$\left| k - \frac{\lfloor \frac{d}{q} \rceil q}{q_0} \right| < \frac{1}{2}. \tag{3}$$

Therefore, re-quantizing $\lfloor \frac{d}{q} \rceil q$ with q_0 yields back k and hence d.

Our methods of watermarking, adaptive attacks, and detection will apply the above DCT Invariant Property several rounds. Figure 3 (a) sketches the pipeline of our watermarking method. In our method, the secret information S is divided into three parts: the secret key, the watermark bits, and a special reference switch bit. The secret key is used for DCT coefficient selector to determine the embedding positions. The reference switch bit along with the key is used to determine a reference bit. With reference to the reference bit, the watermark bits are first differentially encoded and then embedded into the Least Significant Bit (LSB) of the selected DCT coefficients after the latter is further quantized with the respective quantization step size from the quantization table corresponding to $QF = 50$. Formally, for each 8×8 JPEG block, the watermark embedding process consists of the following 4 steps:

DCT coefficient selector The first step is to randomly select 5 DCT coefficients from 16 possible embedding positions. This selector requires a key of length $\lceil \log_2 \binom{16}{5} \rceil$ for each block. Among the 5 selected DCT coefficients, the first one is used as a reference to determine, along with the special reference switch bit, the reference bit. The other 4 selected DCT coefficients are used for watermark embedding, namely the embedding positions. There are 4 watermarking bits per block, one for each embedding position. Therefore, the length of S per 8×8 JPEG block is $\lceil \log_2 \binom{16}{5} \rceil + 5$.

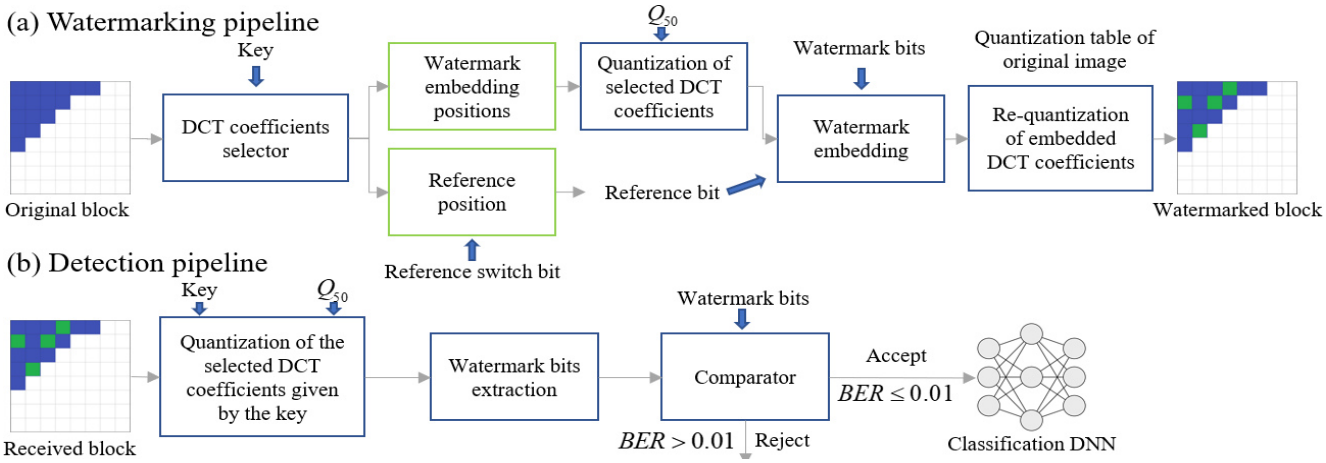

Figure 3. Illustration of the pipelines of watermarking and detection methods. The blue part of the original block represents the 16 possible embedding positions. The green part of the watermarked block represents the selected watermark embedding positions.

Quantization of selected DCT coefficients Denote the DCT coefficient at the watermark embedding position i as $d(i)$. As the second step, we quantize $d(i)$ with the quantization step size $Q_{50}(i)$ from the quantization table corresponding to $QF = 50$ as follows:

$$D_{50}(i) = \lfloor \frac{d(i)}{Q_{50}(i)} \rceil. \tag{4}$$

Watermark embedding Given a watermark bit w to be embedded into the embedding position i, we differentially encode w with respect to the reference bit to derive an embedding value E. The bit w is then embedded into the position i by embedding E into the LSB of $D_{50}(i)$ as follows:

$$D_{50}^{wm}(i) = 2\lfloor D_{50}(i)/2 \rfloor + E, \tag{5}$$

where $D_{50}^{wm}(i)$ is the embedded DCT coefficient integer.

The calculation of E is another important operation in our method. Let $d(j)$ be the selected reference DCT coefficient. To determine the reference bit r for this block, we first quantize $d(j)$ with the quantization step size $Q_{50}(j)$ and then select r to be the second or third last bit of the quantized coefficient integer, depending on whether the special reference switch bit s is 0 or 1. Formally,

$$r = \lfloor \lfloor \frac{d(j)}{Q_{50}(j)} \rceil / 2^{s+1} \rfloor \bmod 2. \tag{6}$$

Finally, the embedding value E for the watermark bit w is determined by

$$E = r \oplus w. \tag{7}$$

The underlying rationale for embedding E rather than w directly is to prevent the adversary from directly accessing the watermark bits through the LSB of $D_{50}^{wm}(i)$. Otherwise, even though the selected watermark embedding positions are not known to the adversary, the adversary can iterate over all possible embedding positions and keep the corresponding watermark bits consistent with the watermarked image so as to bypass the detector without significantly deviating from desired adversarial examples (see Section 4.4 for more details).

Re-quantization of embedded DCT coefficient Finally, the embedded DCT coefficient needs to be re-quantized using the quantization table of the original image so as to keep the consistency of quantization. The watermarked and re-quantized DCT coefficient with respect to the original quantization step size $D_{ori}^{wm}(i)$ can be obtained from the following process:

$$D_{ori}^{wm}(i) = \lfloor D_{50}^{wm}(i) * Q_{50}(i)/Q_{ori}(i) \rceil, \tag{8}$$

where $Q_{ori}(i)$ is the quantization step size at position i in the quantization table of original image. Here we assume that the original quantization step size $Q_{ori}(i)$ is strictly less than $Q_{50}(i)$, which is the case in many applications.

Repeat this procedure for all JPEG blocks in the Luminance channel. The resulting image will be the watermarked image. It took a single 4.5 GHz CPU approximately 0.3 second to compute (in Python) an watermarked image for ImageNet dataset.

4.3. Detection Method

Figure 3 (b) illustrates the pipeline of our detection method. Given a received block, which could be a watermarked block, a watermarked and attacked block, or a watermarked and JPEG re-compressed block, the detection method has three main steps:

Quantization with Q_{50} Determine the 5 selected embedding positions from the key. Quantize the DCT coefficients at those determined positions with Q_{50} to compute the respective $\hat{D}_{50}(i)$, in a way similar to Eq. (4). If the received block is not attacked, nor re-compressed, it follows

from the DCT Invariant Property that $\hat{D}_{50}(i)$ will be equal to $D_{50}^{wm}(i)$.

Watermark bits extraction For each watermark embedding position, take the LSB of $\hat{D}_{50}(i)$ as the estimation $\hat{E}$ of E. Also, compute the estimation $\hat{r}$ of the reference bit r from the reference position according to Eq. (6) with $d(j)$ replaced by $\hat{d}(j)$. The extracted watermark bit $\hat{w}$ corresponding to the watermark bit w is then computed as

$$\hat{w} = \hat{E} \oplus \hat{r}. \tag{9}$$

Comparator For all the watermark embedding positions, we compare the extracted watermark bits with the original watermark bits and compute the Bit Error Rate (BER). The BER represents the percentage of embedded DCT coefficients that are significantly distorted and a larger BER means more distortion is added to the watermarked image. We empirically set a BER threshold of 0.01 to distinguish attacked images from benign images. The received image will be accepted by the detector only if it yields a $BER \leq 0.01$.

Finally, after the received image is accepted, its quantized version after the step 1 above is presented to the classification DNN.

4.4. Adversary Design

Regular adversaries usually work with valid RGB images with fixed image size. That is, input images to a regular adversarial attack algorithm normally take integer-valued pixel intensities and also have their size equal to the input size of the classification DNN. In order for regular adversaries to be able to work within our proposed framework, certain modifications are necessary. Below we first describe how to modify regular adversaries to their advantage so that they can work with JPEG images with various resolutions, and then further extend them to attack our detection strategy itself.

4.4.1 Modifications of Regular Adversaries

A regular adversary is modified via the following key steps:

No pixel rounding in JPEG decoding A regular adversary is modified to take JPEG images directly as their inputs. However, in the process of decoding a JPEG image into its RGB pixel intensities, real-valued pixel values will be kept without any rounding. This avoids possible damages caused by rounding on watermark bits.

Adaptation to various resolutions In our framework, it is necessary for the adversary to provide adversarial examples with the same size as the image to be attacked. To this end, we integrate the resizing process into the classification model as the front layer, which resizes the image to the model's input size. The adversary can then directly add adversarial perturbations into the image through either gradient-based attacks or optimization-based attacks for the integrated model. The resulting attacked images are adversarial examples for the integrated model, and after resizing, are also adversarial examples for the original model.

JPEG encoding Finally, the attacked images, after being JPEG compressed using the same quantization table as in the original image, are adversarial examples produced by the modified regular adversary.

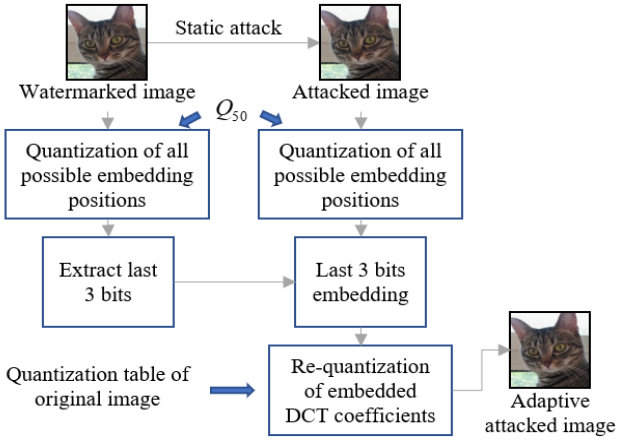

Figure 4. Illustration of the last step in a Type 2 adaptive adversary, which is a replacement of the JPEG encoding step in a modified regular adversary.

4.4.2 Adaptive Adversaries

We propose two adaptive adversaries to weaken our watermarking-based defense.

Type 1 adaptive adversary The JPEG encoding step in a modified regular adversary may weaken its attack strength. To eliminate the negative impact of JPEG encoding on attack strength, we apply JPEG-resistant method proposed by Shin *et al*. [25] to strengthen modified regular adversaries including PGD, FGSM, and CW-l2, resulting our Type 1 adaptive adversaries.

Type 2 adaptive adversary Taking the advantage of the complete knowledge of our watermarking and detection methods, this type of adaptive adversary is designed to completely bypass the detector. It replaces the last JPEG encoding step in a modified regular adversary by the pipeline shown in Figure 4. The DCT Invariant Property guarantees that after both the watermarked image and the adaptive attacked image are quantized with quantization table Q_{50}, the quantized DCT coefficient integer of the DCT coefficient in the adaptive attacked image and its counterpart in the watermarked image have the same last three bits at each possible embedded position. As we shall see later, although this type of adaptive adversary completely bypasses the detector, it

does not necessarily cause harm to the DNN to be secured.

5. Experiment setup

We evaluated our proposed defense against adversarial examples on a subset of ImageNet ILSVRC 2012 validation dataset, which was formed by randomly choosing 1,000 images from the whole validation dataset. All selected images are classified correctly before and after watermarking by *ResNet50V2* [14]. (Otherwise, a new image would be selected and tested until this condition was satisfied.) Our adversarial images were produced by attacking *ResNet50V2*, a pre-trained DNN obtained from *TensorFlow* [1].

Three representative targeted adversarial attack methods FGSM, PGD and CW-l2 were selected. They were implemented with the reference implementations from the *CleverHans* package [20], which were slightly modified to accommodate the modifications mentioned in Section 4.4. The targets for targeted attacks were randomly selected. The parameters selected for these attacks are described below:

- For FGSM and PGD, the adversarial perturbations are computed subject to an L_∞ constraint and the parameter ϵ controls the magnitude of maximum perturbation per pixel. To evaluate our framework under different perturbation levels, we employed targeted FGSM attacks with $\epsilon = 2, 4$ and 8, as well as targeted PGD attacks with $\epsilon = 8$. The selected parameters are commonly used values in other studies [8].
- For CW-L2, the adversarial perturbations are optimized under L_2 constraint. Its hyper-parameter κ specifies the confidence that the adversarial image is misclassified by the target DNN, and also controls the amount of perturbations. The smaller κ, the smaller perturbations. Since small perturbations are difficult to be detected, we employed targeted CW-l2 attacks with $\kappa = 0$ to evaluate our proposed defense strategy in the worst case scenario.

6. Results

In this section, we firstly show the effect of watermarking on classification accuracy and image quality, followed by its robustness to JPEG re-compression. Then, we evaluate the effectiveness of our defense against a wide range of adversaries. Note that in view of the DCT Invariant Property, the false positive rate is guaranteed to be 0.

6.1. Classification accuracy and PSNR

Table 1 shows the top-1 and top-5 accuracy before and after watermarking of ResNet50V2, MobileNetV2, InceptionV3 over the entire ImageNet ILSVRC 2012 validation dataset. The impact of watermarking on classification accuracy is indeed insignificant with, on average, 0.63% and 0.49% degradation in top 1 and top 5 accuracy, respectively.

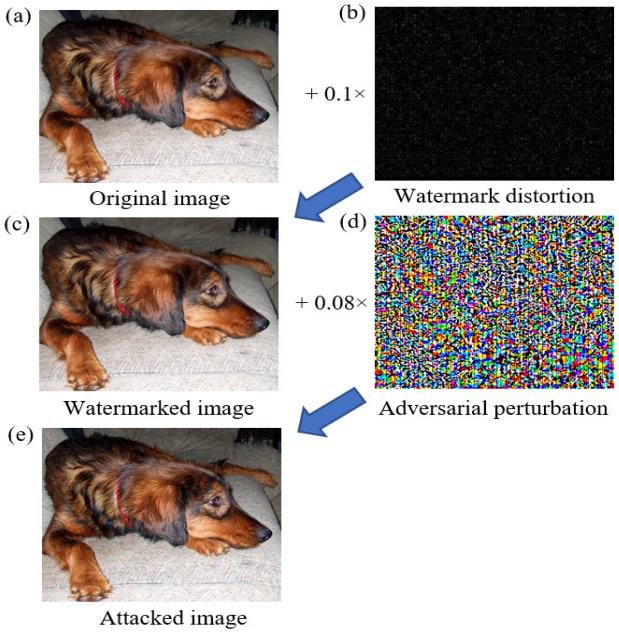

Figure 5. Illustration of watermarking and adversarial perturbation: (a) the original image; (b) watermark distortion (amplified 10 times); (c) the watermarked image; (d) the adversarial perturbation generated by FGSM with $\epsilon = 8$; and (e)the FGSM attacked image.

DNN	Top1	Top1 wm	Top5	Top5 wm
ResNet50V2	67.00%	66.51%	87.81%	87.43%
MobileNetV2	70.85%	69.63%	89.80%	89.01%
InceptionV3	76.85%	76.66%	93.30%	93.00%

Table 1. Top-1 and Top-5 accuracy before and after watermarking for three pre-trained DNNs from *TensorFlow* [1]. Note that the classification accuracy may be different from that reported by the original work due to different pre-processing methods.

The PSNR between the original and watermarked images is found to be 39.34 ± 1.13 dB. As shown in Figure 5, the watermark distortion is imperceptible, and significantly less than the adversarial perturbation.

6.2. Robustness to JPEG re-compression

To evaluate watermarking robustness to JPEG re-compression, we conducted an experiment where watermarked images are first compressed by multiple rounds of JPEG re-compression with each QF randomly selected from $[50, 100)$, and then sent to the detector. Table 2 reports the respective average BER of the detector in each case. It is clear from Table 2 that our watermarking method is indeed very robust to one or two rounds of high quality JPEG re-compression. The results in Table 2 also justify empirically our choice of BER threshold 0.01.

Rounds	1	2	3	5
Average BER	0.00067	0.0082	0.021	0.038

Table 2. Average BER for different rounds of JPEG re-compression.

Metric	FGSM $\epsilon = 2$	FGSM $\epsilon = 4$	FGSM $\epsilon = 8$	PGD $\epsilon = 8$
Detection rate	99.7%	100.0%	100.0%	100.0%
EFNR	0.2%	0.0%	0.0%	0.0%

Table 3. Detection rate and effective false negative rate in the case of static white-box FGSM and PGD attacks.

Metric	FGSM $\epsilon = 2$	FGSM $\epsilon = 4$	FGSM $\epsilon = 8$	PGD $\epsilon = 8$
Detection rate	99.8%	100.0%	100.0%	100.0%
EFNR	0.1%	0.0%	0.0%	0.0%
Detection rate	0.0%	0.0%	0.0%	0.0%
EFNR	0.1%	0.0%	0.4%	1.5%

Table 4. Detection rate and effective false negative rate in the case of adaptive white-box FGSM and PGD attacks: top for type 1 adaptive adversary; bottom for type 2 adaptive adversary.

6.3. Combating FGSM and PGD attacks

Table 3 and Table 4 show the detection rate and effective false negative rate in the case of static FGSM and PGD attacks, and in the case of adaptive FGSM and PGD attacks, respectively. Clearly, the detector can effectively detect adversarial perturbations introduced by static and type 1 adaptive FGSM and PGD attacks. Although type 2 adaptive FGSM and PGD can bypass the detector completely by design, they are nonetheless harmless to the subsequent classification DNN with near zero EFNR. The quantization process with Q_{50} along with forcing the last three bits at each possible embedding position to be the same as those of the counterpart in the watermarked image essentially undoes the adversarial perturbation.

6.4. Combating CW-l2 attack

CW-l2 attack [5] provides strong adversarial examples with high confidence under a tight perturbation budget. The light perturbation increased the difficulty of detecting it. However, as suggested in [13, 8], adversarial perturbation generated by CW-l2 method is fragile to JPEG compression. The results for both static and adaptive white-box CW-l2 attacks are shown in Table 5. The relatively-low detection rate of static CW-l2 attack suggests that the watermarked coefficients are barely distorted after quantization. On the other hand, converting adversarial examples to JPEG format also significantly decreases the effectiveness of the static attack, resulting in a low effective false negative rate. With the JPEG-resistant feature, type 1 adaptive CW-l2 yielded a higher effective false negative rate at 25.3%. It also increased the distortion required and resulted in a higher detection rate.

	Static	Type 1	Type 2
Detection rate	34.1%	38.4%	0%
EFNR	4.7%	25.3%	0.0%

Table 5. Detection rate and EFNR in the case of static, type 1 adaptive, and type 2 adaptive CW-l2 attacks.

BER threshold	0.0025	0.005	0.0075	0.01
Detection rate	85.1%	67.5%	54.6%	38.4%
EFNR	5.8%	12.3%	16.7%	25.3%

Table 6. Detection rate and effective false negative rate in the case of type 1 adaptive CW-l2 attacks for different BER thresholds.

To combat CW-l2 attack, an effective way is to decrease the BER threshold of the detector. Table 6 shows the detection rate and effective false negative rate in the case of type 1 adaptive CW-l2 attacks for different BER thresholds. When the BER threshold is set to 0.0025, the detection rate increases significantly from 38.4% to 85.1%, whereas the effective false negative rate decreases significantly from 25.3% from 5.8%. The improved performance is at the cost of watermarking robustness to multiple rounds of high quality JPEG re-compression. With the BER threshold at 0.0025, our watermarking method is very robust to only one round of JPEG re-compression.

7. Conclusion

In this paper, we have proposed a novel and practical watermarking-based framework for protecting image task related DNNs against adversarial attacks. The framework consists of a watermark encoder, a possible adversary, and a detector. Specific methods of watermarking and detection have been presented. It has been shown by experiment that the framework is effective against a wide range of advanced attacks (static and adaptive), achieving a near zero (effective) false negative rate with the guaranteed zero false positive rate. At the same time, the impact of watermarking on classification accuracy of DNNs is insignificant.

Acknowledgement

This work was supported in part by the Natural Sciences and Engineering Research Council of Canada under Grant RGPIN203035-16, and by the Canada Research Chairs Program. We would like to thank Hossam Amer and Bin Chen for some helpful discussions, and Hossam Amer for sharing his JPEG encoding/decoding related source code.

References

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*, pages 265–283, 2016.

[2] Moustafa Alzantot, Yash Sharma, Supriyo Chakraborty, Huan Zhang, Cho-Jui Hsieh, and Mani B Srivastava. Genattack: Practical black-box attacks with gradient-free optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 1111–1119, 2019.

[3] Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion attacks against machine learning at test time. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 387–402. Springer, 2013.

[4] Nicholas Carlini and David Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pages 3–14, 2017.

[5] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. IEEE, 2017.

[6] Jianbo Chen and Michael I Jordan. Boundary attack++: Query-efficient decision-based adversarial attack. *arXiv preprint arXiv:1904.02144*, 2(7), 2019.

[7] Ingemar Cox, Matthew Miller, Jeffrey Bloom, Jessica Fridrich, and Ton Kalker. *Digital watermarking and steganography*. Morgan kaufmann, 2007.

[8] Nilaksh Das, Madhuri Shanbhogue, Shang-Tse Chen, Fred Hohman, Siwei Li, Li Chen, Michael E Kounavis, and Duen Horng Chau. Shield: Fast, practical defense and vaccination for deep learning using jpeg compression. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 196–204, 2018.

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[10] Anna Egorova and Victor Fedoseev. Semi-fragile watermarking for jpeg image authentication: A comparative study. In *2019 7th International Symposium on Digital Forensics and Security (ISDFS)*, pages 1–6. IEEE, 2019.

[11] Justin Gilmer, Luke Metz, Fartash Faghri, Samuel S Schoenholz, Maithra Raghu, Martin Wattenberg, and Ian Goodfellow. Adversarial spheres. *arXiv preprint arXiv:1801.02774*, 2018.

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

[13] Chuan Guo, Mayank Rana, Moustapha Cisse, and Laurens Van Der Maaten. Countering adversarial images using input transformations. *arXiv preprint arXiv:1711.00117*, 2017.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer, 2016.

[15] Ching-Yung Lin and Shih-Fu Chang. Semifragile watermarking for authenticating jpeg visual content. In *Security and Watermarking of Multimedia Contents II*, volume 3971, pages 140–151. International Society for Optics and Photonics, 2000.

[16] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[17] Dongyu Meng and Hao Chen. Magnet: a two-pronged defense against adversarial examples. In *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, pages 135–147, 2017.

[18] Jan Hendrik Metzen, Tim Genewein, Volker Fischer, and Bastian Bischoff. On detecting adversarial perturbations. *arXiv preprint arXiv:1702.04267*, 2017.

[19] Ridvan Ozdemir and Mehmet Koc. A quality control application on a smart factory prototype using deep learning methods. In *2019 IEEE 14th International Conference on Computer Sciences and Information Technologies (CSIT)*, volume 1, pages 46–49. IEEE, 2019.

[20] Nicolas Papernot, Fartash Faghri, Nicholas Carlini, Ian Goodfellow, Reuben Feinman, Alexey Kurakin, Cihang Xie, Yash Sharma, Tom Brown, Aurko Roy, Alexander Matyasko, Vahid Behzadan, Karen Hambardzumyan, Zhishuai Zhang, Yi-Lin Juang, Zhi Li, Ryan Sheatsley, Abhibhav Garg, Jonathan Uesato, Willi Gierke, Yinpeng Dong, David Berthelot, Paul Hendricks, Jonas Rauber, and Rujun Long. Technical report on the cleverhans v2.1.0 adversarial examples library. *arXiv preprint arXiv:1610.00768*, 2018.

[21] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia conference on computer and communications security*, pages 506–519, 2017.

[22] Pouya Samangouei, Maya Kabkab, and Rama Chellappa. Defense-gan: Protecting classifiers against adversarial attacks using generative models. *arXiv preprint arXiv:1805.06605*, 2018.

[23] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[24] Yash Sharma, Gavin Weiguang Ding, and Marcus Brubaker. On the effectiveness of low frequency perturbations. *arXiv preprint arXiv:1903.00073*, 2019.

[25] Richard Shin and Dawn Song. Jpeg-resistant adversarial images. In *NIPS 2017 Workshop on Machine Learning and Computer Security*, volume 1, 2017.

[26] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE con-*

ference on computer vision and pattern recognition, pages 2818–2826, 2016.

[27] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[28] Yusuke Tsuzuku and Issei Sato. On the structural sensitivity of deep convolutional networks to the directions of fourier basis functions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 51–60, 2019.

[29] Javier Villalba-Diez, Daniel Schmidt, Roman Gevers, Joaquín Ordieres-Meré, Martin Buchwitz, and Wanja Wellbrock. Deep learning for industrial computer vision quality control in the printing industry 4.0. *Sensors*, 19(18):3987, 2019.