Are Deep Neural Networks good for blind image watermarking?

Vedran Vukotić, Vivien Chappelier, Teddy Furon

Are Deep Neural Networks good for blind image watermarking?

Vedran Vukotić
Lamark
name.surname@lamark.fr

Vivien Chappelier
Lamark
name.surname@lamark.fr

Teddy Furon
Univ Rennes, Inria, CNRS, IRISA
name.surname@inria.fr

Abstract

Image watermarking is usually decomposed into three steps: i) some features are extracted from an image, ii) they are modified to embed the watermark, iii) and they are projected back into the image space while avoiding the creation of visual artefacts. The feature extraction is usually based on a classical image representation given by the Discrete Wavelet Transform or the Discrete Cosine Transform for instance. These transformations need a very accurate synchronisation and usually rely on various registration mechanisms for that purpose.

This paper investigates a new family of transformation based on Deep Learning networks. Motivations come from the Computer Vision literature which has demonstrated the robustness of these features against light geometric distortions. Also, adversarial sample literature provides means to implement the inverse transform needed in the third step.

This paper shows that this approach is feasible as it yields a good quality of the watermarked images and an intrinsic robustness.

1. Introduction

Deep Learning (DL) has completely revolutionized the field of Computer Vision. It started with image classification [14] and is now spreading to any task of Computer Vision: object recognition [21], instance search [25], localization [11], similar image retrieval [20, 10] ... The transfert property is underlying this versatility: A DL network trained on some supervised task (*e.g.* classification) has been shown to perform well on other applications, even non-supervised tasks like similarity search.

A DL network comprises several layers. While the last layer provides a vector of probabilities that the input image belongs to the classes, the internal auxiliary data (the output of the previous layers) also happen to contain relevant information about the input image. Other tasks can indeed tap these data. Many Computer Vision works take a trained network off the shelf, keep the first layers as is, and only re-train the deepest layers for their specific task [20].

Following this trend, state-of-the-art image search algorithms tap the output of an internal layer and use them to derive a global descriptor of the image [20, 10]. This way, finding similar images boils down to finding close vectors in an Euclidean high dimensional space, where efficient fast search engines exist. It has been shown that these global descriptors are compact and discriminative while at the same time robust to valuemetric (*e.g.* JPEG, noise, filtering) and light geometric (*e.g.* cropping, rotation, scaling) distortions [12].

1.1. Problem formulation

This paper investigates whether this approach is also suitable for watermarking, with the hope of benefiting from this apparent robustness. This paper only considers zero-bit watermarking. The layers of a DL network play the role of the extraction function yielding a vector to be watermarked. This raises the following challenges:

- How to invert this highly non-linear extraction function? Once the feature vector is watermarked, how to map it back into the image space?
- How to take into account a perceptual model ensuring the quality of the watermarked image?
- How to guarantee a required false positive level as there is no probabilistic modeling of these features?

1.2. Prior works

The main trend dealing with DL and watermarking is actually the protection of DL networks. It aims at proving the ownership of a network by embedding a watermark into the learned parameters of the model [19, 6].

The connection between machine learning and *image* watermarking is surprisingly old. Papers [26, 13] protect images with a classical watermarking technique and a neural network is only used at the decoding side in place of a maximum likelihood decoder based on a statistical model, which may not be accurate. J. Stolarek and P. Lipiński learn a transformation [23] as proposed in this paper. However,

this transform is dedicated to one specific host image. Paper [18] has a similar position but fails guaranteeing a probability of false alarm.

We also mention paper [7] which makes a very good comparison between attacks in watermarking and attacks on DL networks (*a.k.a.* adversarial sample - see Sect. 3.1).

2. Zero-bit Watermarking

Our scheme belongs to the TEMIT approach[1]: Transform, EMbed, Inverse Transform. We denote by $\mathsf{Tr}(\cdot)$ the transformation extracting a feature vector, so-called the host signal $\mathbf{x}_o = \mathsf{Tr}(\mathcal{I}_o)$ of dimension n, from an image $\mathcal{I}_o$. The embedding modifies it into $\mathbf{x}_m = \mathsf{Emb}(\mathbf{x}_o)$. This function implicitly depends on a secret key. For simplicity, the paper focuses on a zero-bit watermarking scheme [16, 8, 4]: we hide the presence of a secret mark, not a message. The watermarked image is given by $\mathcal{I}_m = \mathcal{I}_o + \mathsf{Tr}^{-1}(\mathbf{x}_m - \mathbf{x}_o)$, where Tr^{-1} is the inverse transform function.

At the detection side, the image under scrutiny is $\mathcal{I}_a$ with $\mathbf{x}_a = \mathsf{Tr}(\mathcal{I}_a)$. It is deemed as a watermarked image if $\mathbf{x}_a$ belongs to the acceptance region $\mathcal{D} \subset \mathbb{R}^n$. The use of a secret key is implicit in these notations.

2.1. The hypercone detector

This paper focuses on one specific zero-bit watermarking scheme: the hypercone detector [16]. This scheme is provably good [4] and the detector is blind and oblivious to the watermark, host and noise powers.

The acceptance region $\mathcal{D}$ is a dual hypercone of axis $\mathbf{a} \in \mathbb{R}^n$ ($\|\mathbf{a}\| = 1$) and half angle $0 < \theta < \pi/2$ defined as:

$$\mathcal{D} := \{\mathbf{x} \in \mathbb{R}^n : |\mathbf{x}^\top \mathbf{a}| > \|\mathbf{x}\| \cos(\theta)\}. \tag{1}$$

Vector $\mathbf{a}$ plays the role of the secret key.

We now define the following function $R(\cdot) : \mathbb{R}^n \mapsto \mathbb{R}$:

$$R(\mathbf{x}) = (\mathbf{x}^\top \mathbf{a})^2 \rho(\theta) - \|\mathbf{x}\|^2, \tag{2}$$

with $\rho(\theta) := (\tan^2(\theta) + 1)$. This quantity is negative when $\mathbf{x} \notin \mathcal{D}$, null when $\mathbf{x}$ lies on the boundary of $\mathcal{D}$, and positive inside the dual hypercone. Indeed, when $R(\mathbf{x}) > 0$, this quantity is the amount of power of Gaussian white noise to add onto $\mathbf{x}$ in order to push it outside the hypercone with high probability [4].

This gives a rationale for watermarking vectors at the embedding side. We push the host vector $\mathbf{x}_o$ to a location $\mathbf{x}_m$ deep inside $\mathcal{D}$ s.t. $R(\mathbf{x}_m)$ is as big as possible. This provably increases the robustness against a white Gaussian noise addition in the feature space.

[1] a wording coined by Ton Kalker.

2.2. Linear and invertible extraction function

The hypercone detector is at the core of some image watermarking techniques [8] where the extraction function is (almost) a $n \times |I|$ Parseval tight frame T (*e.g.* selection of some DCT or DWT coefficients). Once the embedding modifies $\mathbf{x}_o$ in $\mathbf{x}_m$, the watermarked image is obtained as

$$\mathcal{I}_m = \mathcal{I}_o + T^\top(\mathbf{x}_m - \mathbf{x}_o). \tag{3}$$

A constraint on the image Euclidean distance $\|\mathcal{I}_m - \mathcal{I}_o\|_2 \leq C$ (*e.g.* expressed in terms of MSE or PSNR) is ensured if $\|\mathbf{x}_m - \mathbf{x}_o\|_2 \leq C$. The embedding then amounts to solve the following problem:

$$\mathbf{x}_m = \arg \max_{\mathbf{x}:\|\mathbf{x}-\mathbf{x}_o\| \leq C} R(\mathbf{x}). \tag{4}$$

It is known that $\mathbf{x}_m$ belongs to the 2D-hyperplane containing the origin, $\mathbf{x}_o$, and $\mathbf{u}$ [5, 16], so that (4) boils down to a line search over angle $\beta \in [0, \pi/2]$ defining:

$$\mathbf{x} = \mathbf{x}_o + C\cos(\beta)\mathbf{u} - C\sin(\beta)\mathbf{v}, \tag{5}$$

with $(\mathbf{u}, \mathbf{v})$ a basis of this hyperplane:

$$\mathbf{u} := \mathsf{sign}(\mathbf{x}_o^\top \mathbf{a})\mathbf{a}, \quad \mathbf{v} := \frac{\mathbf{x}_o - (\mathbf{x}_o^\top \mathbf{u})\mathbf{u}}{\|\mathbf{x}_o - (\mathbf{x}_o^\top \mathbf{u})\mathbf{u}\|}. \tag{6}$$

Vectors $\mathbf{x}_o$ and $\mathbf{a}$ are statistically independent, so that for large n, $\mathbf{x}_o^\top \mathbf{a} \approx 0$ which yields the close form solution [4]:

$$\beta^\star = \arccos \sqrt{1 - \|\mathbf{x}_o\|^2 C^{-2} \cos^4 \theta}, \tag{7}$$

if $C \geq \|\mathbf{x}_o\| \cos^2 \theta$, otherwise $\mathbf{x}_o$ cannot be watermarked (*i.e.* pushed into region $\mathcal{D}$) because C is too small.

The embedding is thus given by a close form solution or a simple line search over β. This is thanks to an invertible extraction function Tr preserving the Euclidean distance.

3. Deep learning feature extraction

This section considers a neural network used as image classifier over n_c classes. It is denoted by function F that computes a vector from an image: $\mathbf{p} = \mathsf{F}(\mathcal{I}) \in \mathbb{R}^{n_c}$. This function is decomposed as $\mathsf{F} = \mathsf{S} \circ \mathsf{N}$, where S is the softmax function ensuring that $\mathbf{p}$ is a probability distribution (*i.e.* $\sum_{k=1}^{n_c} p_k = 1$, and $0 \leq p_k, \forall k \in \{1, \ldots, n_c\}$). Function N is the network per se, a composition of ℓ_C convolutionnal layers and ℓ_F fully connected layers. Each layer is a non-linear process as it encompasses an activation function such as the well-known ReLU.

Once the parameters of these layers have been learned from a training set, the network predicts classes of images. The output of the softmax is a vector $\mathbf{p} = \mathsf{F}(\mathcal{I})$ of estimated probabilities $p_k = \hat{P}(k|\mathcal{I})$ that content $\mathcal{I}$ belongs to class k. In the end, the class of an image is given by

$$\mathsf{C}(\mathcal{I}) = \arg \max_{1 \leq k \leq n_c} p_k. \tag{8}$$

3.1. Adversarial samples

Adversarial sampling is a recent trend showing that DL classifiers can easily be fooled [17, 3]. The attack forges an image $\mathcal{I}_a$ visually similar to an original image $\mathcal{I}_o$ but with a different prediction $\mathsf{C}(\mathcal{I}_a) \neq \mathsf{C}(\mathcal{I}_o)$.

In a targeted attack, a specific class is given, say k_t, and an original image $\mathcal{I}_o$ not belonging to this class. The attack finds $\mathcal{I}_a$ as close as possible to $\mathcal{I}_o$ s.t. $\mathsf{C}(\mathcal{I}_a) = k_t$:

$$\mathcal{I}_a = \arg \min_{\mathcal{I}:\mathsf{C}(\mathcal{I})=k_t} \mathsf{Dist}(\mathcal{I}, \mathcal{I}_o), \tag{9}$$

with Dist a measure of distortion (usually, the L_ℓ-norm of $(\mathcal{I} - \mathcal{I}_o)$ with $\ell \in \{0, 1, 2, +\infty\}$). In a white box scenario, the attacker knows the network architecture and parameters.

Problem (9) is impossible to be solved as the region of images classified as k_t is unknown. It is replaced by

$$\mathcal{I}_a = \arg\min \mathsf{J}_\lambda(\mathcal{I}) \tag{10}$$

$$\mathsf{J}_\lambda(\mathcal{I}) := \mathsf{Loss}(\mathcal{I}, k_t) + \lambda.\mathsf{Dist}(\mathcal{I}, \mathcal{I}_o). \tag{11}$$

where $\lambda \in \mathbb{R}^+$ and Loss is a loss function measuring how far $\mathcal{I}$ is from being classified as class k_t. For instance, the loss is a divergence (*e.g.* Cross-Entropy) between the predicted probability distribution $\mathsf{F}(\mathcal{I})$ and the targeted $\mathbf{p}^\star$, *i.e.* $p^\star_k = \delta_{k_t}(k)$, or more simply:

$$\mathsf{Loss}(\mathcal{I}, k_t) = |\max_{k \neq k_t}(\mathsf{F}(\mathcal{I})_k) - \mathsf{F}(\mathcal{I})_{k_t}|_+, \tag{12}$$

with $|a|_+ = a$ if $a > 0$ and 0 otherwise. This way

$$\mathsf{C}(\mathcal{I}) = k_t \Leftrightarrow \mathsf{Loss}(\mathcal{I}, k_t) = 0. \tag{13}$$

3.2. Practical solutions

The minimization problem (10) implicitly uses the domain of images of the same size $h \times l$ and same number of channels c as the original image $\mathcal{I}$, say $\{0, 1, \ldots, 255\}^{h \times l \times c}$. A relaxation looks for a solution over the continuous set $[0, 255]^{h \times l \times c}$. Quantization onto integers is applied on the continuous solution to obtained an image. This box-constrained optimization often uses a differentiable mapper m which is a monotonic bijection from $\mathbb{R}$ to $[0, 255]$, *e.g.* $\mathsf{m}(x) = 255(\tanh(x) + 1)/2$ as in [3]. This changes (9) into a unconstrained minimization of $\mathsf{J}_\lambda(\mathsf{m}(X))$ over $X \in \mathbb{R}^{h \times l \times c}$.

Practical attacks are mostly based on gradient descent [9, 15]. One can back-propagate the computation of the gradient through the layers of the network. The attack initializes a random starting point $X^{(0)}$ (around $\mathsf{m}^{-1}(\mathcal{I})$) and iteratively computes

$$X^{(i+1)} = X^{(i)} - \eta \nabla \mathsf{J}_\lambda(\mathsf{m}(X^{(i)})). \tag{14}$$

It stops when the objective function no longer decreases substantially. This gradient descent converges to a local minimum. Several rounds with different initialization are competing. Advanced numerical solvers have been also used in the literature, *e.g.* ADAM [3]. The main difficulty is to set λ to a well chosen constant. Theoretically, there exists a range of λ values where the solution of (10) coincides with the solution of (9). In practice, this range is unknown.

Simpler attacks start with a given distortion budget, $\mathsf{Dist}(\mathcal{I}_a, \mathcal{I}_o) \leq C$, and set $\lambda = 0$. The gradient descent starts at the original image, *i.e.* $X^{(0)} = \mathsf{m}^{-1}(\mathcal{I}_o)$ and iterates until either $\mathsf{C}(\mathsf{m}(X^{(j)})) = k_t$ or $\mathsf{Dist}(\mathsf{m}(X^{(j)}), \mathcal{I}_o) > C$. In the latter case, the attack failed finding an adversarial sample within the distortion budget.

4. Application to zero-bit watermarking

The key idea of this paper is i) to use the network N or part of it (*i.e.* the first layers) as the extraction function Tr, ii) to modify the extracted vector as explained in Sect. 2, iii) to use an adversarial sample mechanism of Sect. 3 as Tr^{-1} in order to put back the marked vector into the image. This raises several issues cleared in this section.

4.1. Need of a locality transform

Our proposed system can function with any neural architecture. Denote by E the set of the first layers used for extracting an embedding of an image: $\mathbf{e} = \mathsf{E}(\mathcal{I}) \in \mathbb{R}^n$. Yet, the representational coordinate system of the embedding space does not depict the feasible locations. A neural network never provides a zero-mean isotropic embedding $\mathbf{e}$. The usual culprit being the asymmetrical activation functions, such as ReLU, that have a $\mathbb{R}_{\geq 0}$ codomain.

This rises the need for a mapping from the original coordinate system to a local coordinate system. We empirically model a local coordinate system by providing images to the DL network and analyzing their embeddings in their representation space. We use two possible mappings:

- **Centering**: $\mathbf{x} = \mathsf{L}(\mathbf{e}) = \mathbf{e} - \bar{\mathbf{e}}$,
- **PCA**: $\mathbf{x} = \mathsf{L}(\mathbf{e}) = \bar{\Sigma}^{-1/2}(\mathbf{e} - \bar{\mathbf{e}})$,

where $\bar{\mathbf{e}}$ is the empirical mean and $\bar{\Sigma}$ is the empirical covariance matrix. The first option preserves the dimensionality, while the second may operate a reduction by keeping the biggest eigenvalues of $\bar{\Sigma}$. In the end, $\mathsf{Tr} = \mathsf{L} \circ \mathsf{E}$.

4.2. False alarm probability

The false alarm probability is usually defined as $P_{\mathsf{fa}} := \mathbb{P}(\mathbf{X} \in \mathcal{D})$ where the random vector $\mathbf{X}$ models the feature vector of an original image. For the hypercone detector, under the weak assumption that $\mathbf{X}$ has an isotropic distribution (*e.g.* a Gaussian white distribution), this probability has a close form expression:

$$P_{\mathsf{fa}} = \mathsf{I}_{\cos^2(\theta)}((n-1)/2, 1/2), \tag{15}$$

where I is the regularized Beta incomplete function.

This assumption does not hold for embeddings provided by the network. Even the empirical centering and whitening doesn't guarantee a distribution *exactly* isotropic.

Instead, we prefer modifying the definition of the probability of false alarm. For a given image under scrutiny, the extracted features are seen as a fixed vector. It is the acceptance region which is random: the axis direction of the hypercone is a random vector $\mathbf{A}$ uniformly distributed on the hypersphere. It happens that this probability of false alarm has the exact same expression as (15). For a required false positive level, (15) is inverted so as to find the corresponding half angle value θ.

Another use of this equation is to compute the p-value: for given $\mathbf{x}$ and $\mathbf{a}$, quantity (15) is computed for a cosine equalling $c = |\mathbf{x}^\top \mathbf{a}|/\|\mathbf{x}\|$. It means that if we were drawing $O(1/p)$ random secret keys $\mathbf{A}$, on expectation, one of them would give a normalized correlation bigger or equal to c.

4.3. The objective function and imposed constraints

A bad idea is to stick too closely to the watermarking described in Sec. 2.2. It amounts to first fix $\mathbf{x}_m$ and then to use the back propagation to craft an image whose feature vector is as close as possible to $\mathbf{x}_m$.

Watermarking in the feature space and crafting the watermarked image are better done jointly by defining:

$$\mathcal{I}_w = \arg \min_{\mathsf{Dist}(\mathcal{I},\mathcal{I}_o)\leq C} \mathsf{Loss}(\mathcal{I}) \quad (16)$$

$$\mathsf{Loss}(\mathcal{I}) = -\mathsf{R}(\mathsf{Tr}(\mathcal{I})), \quad (17)$$

and letting an adversarial sample mechanism solving this optimization problem. The distortion function Dist is for instance the Mean Square Error (expressed in dB as a PSNR).

While minimizing the loss and thus iteratively generating the watermarked image, additional constraints can be applied. For instance, the watermark can be attenuated selectively in the spatial domain by a perceptual mask to make it less perceivable. We use the Structural Similarity SSIM heatmap for that purpose. The watermarking is then performed by a simple gradient descent (see algorithm 1).

5. Experiments

The watermarking system is both architecture and layer agnostic, meaning that any layer can a priori be used for watermarking. This paper considers one specific architecture at different layers, exploring their goodness for watermarking by evaluating robustness and perceptual quality.

5.1. Experimental protocol

The experimental part evaluates our proposed system based on the off-the-shelf *VGG19* [22] convolutional neural network pretrained on *ImageNet*[2]. We perform all the experiments on photos from the test set P "professional" provided as part of the CLIC challenge [1]. This dataset consists of 117 high-resolution professionally taken photos.

[2]available at https://keras.io/applications/#vgg19

Data: input image $\mathcal{I}_o$, watermarking signal $\mathbf{a}$
Result: watermarked image $\mathcal{I}_w$
$i \leftarrow 0$;
$\mathcal{I}_i \leftarrow \mathcal{I}_o$;
repeat
  /* obtain embedding */
  $\mathbf{x} \leftarrow \mathsf{Tr}(\mathcal{I}_i)$;
  $\mathsf{Loss}(\mathcal{I}_i) \leftarrow -R(\mathbf{x})$;
  /* perform one step update */
  $\mathbf{g} \leftarrow \nabla \mathsf{Loss}(\mathcal{I}_i)$;
  $\mathcal{I}_{i+1} \leftarrow \mathcal{I}_i - \eta \mathbf{g}$;
  if *performing SSIM heatmap attenuation* **then**
    $\mathbf{h}_{ssim} \leftarrow SSIM_{heatmap}(\mathcal{I}_o, \mathcal{I}_{i+1})$;
    /* attenuate watermark with psycho-visual heatmap */
    $\mathcal{I}_\delta \leftarrow \mathcal{I}_{i+1} - \mathcal{I}_o$;
    $\mathcal{I}_{i+1} \leftarrow \mathcal{I}_o + \mathbf{h}_{ssim} \otimes \mathcal{I}_\delta$;
  end
  /* compute indicator values */
  Compute $\mathsf{Dist}(\mathcal{I}_o, \mathcal{I}_{i+1})$
  $i \leftarrow i + 1$;
until *stopping criterion met*;
$\mathcal{I}_w \leftarrow \mathcal{I}_i$;

Algorithm 1: Proposed watermarking algorithm.

5.2. Image quality

When the stopping criterion is expressed in terms of PSNR, the quality of the watermarked image is roughly acceptable above 40dB. Fig. 2 illustrates this with the reference Lenna image on its top row. The watermark signal in the image space looks like uniform noise when watermarking lower layers and gets more structured when processing deeper layers, as shown in Fig. 1.

A psycho-visual attenuation as specified in Algorithm 1 improves the quality of the image. Our experiments used the structural similarity (SSIM) heatmap, but any other psycho-visual model (*e.g.* Butteraugli [2]) can be integrated transparently. Figure 2 shows that the watermark is less perceivable on the right column thanks to attenuation in flat regions (skin, hat, wall) for a given $PSNR$ target. This almost does not hurt robustness $\mathsf{R}(\mathsf{Tr}(\mathcal{I}_w))$.

The downside is the increased number of steps (and thus time) necessary to obtain a watermark of the same robustness. Fig. 3 shows a 3D representation of the watermarking process. The first two axis corresponds to the projection

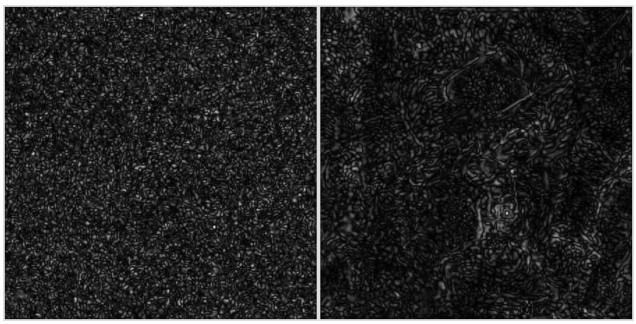

Figure 1. The watermark signal in the image space. Left - *block1_pool*: last pooling layer of the first convolutional block. Right - *fc1*: last fully-connected layer.

Figure 2. Top: $PSNR = 42dB$ and $P_{fa} = 10^{-8}$ without (left) and with (right) structural similarity attenuation. Bottom: 'exaggerated' watermark at $PSNR = 25dB$ without (left) and with (right) structural similarity attenuation.

of $\mathsf{Tr}(\mathcal{I}_i)$ onto the basis $(\mathbf{u}, \mathbf{v})$ (6), while the third component accounts for the energy remaining in the complimentary space. The stopping criterion was met in 10 or 13 iterations, respectively without or with perceptual attenuation.

5.3. From one layer to another

Table 1 gives the $\log_{10} p$-value for the watermarked image and after a rotation of 5°. The first part of the table illustrates the behaviour when centering is used as a locality transform, thus retaining the original dimensionality. The second part of the table illustrates the behaviour with PCA with a reduction to 256 dimensions.

In general, watermarking at deeper layers offers more robustness. Each convolutional–pooling block adding some

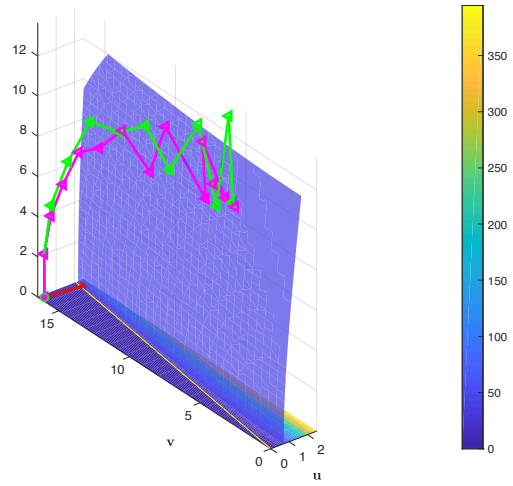

Figure 3. 3D view of the iterations without (green) and with (magenta) the perceptual attenuation. The 3D shape shows the set $\{\mathbf{x} : \mathsf{R}(\mathbf{x}) = c\}$. The red line shows the embedding of Sect. 2.2 providing the same robustness. The yellow line is the boundary of the hypercone. The heatmap of $\mathsf{R}(\mathbf{x})$ on plane $(\mathbf{u}, \mathbf{v})$.

Table 1. Watermarking with different VGG19 layers with the same target $P_{fa} = 1e-6$ and a stopping criterion of $PSNR \leq 42dB$.

Layer name	**Emb. dim.**	**Average $\log_{10} p$-value marked image**	**Average $\log_{10} p$-value rotation 5 °**
Original dimensionality, centering			
block1_pool	802 816	-64.49	-0.68
block2_pool	401 408	-58.32	-1.13
block3_pool	200 704	-52.25	-3.80
block4_pool	100 352	-21.31	-3.81
block5_pool	25 088	-4.95	-2.24
fc1	4 096	-5.27	-4.41
fc2	4 096	-4.11	-1.70
PCA to 256 dimensions			
block1_pool	802 816	-3.65	-0.93
block2_pool	401 408	-12.30	-1.36
block3_pool	200 704	-8.13	-3.20
block4_pool	100 352	-8.19	-2.42
block5_pool	25 088	-4.61	-2.41
fc1	4 096	-5.61	-3.20
fc2	4 096	-5.22	-1.69

invariance to rotation and scaling, the best layer is indeed the first fully-connected layer. From now on, the remaining experiments use this layer.

5.4. Robustness

We evaluate the robustness of the first fully-connected *fc1* layer of VGG19 at the expense of a lower dimension-

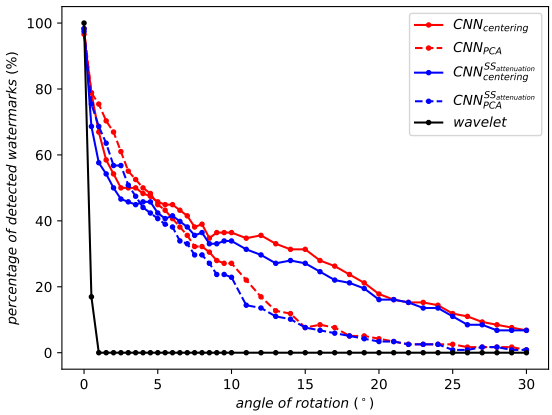

Figure 4. Robustness against rotation.

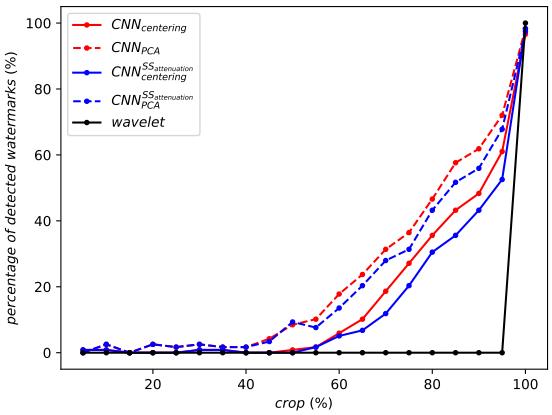

Figure 5. Robustness against cropping.

ality ($n = 4096$). The stopping criterion is defined as $PSNR = 42dB$. To define a baseline, we watermark the same images with the linear wavelet transform as described in Sect. 2.2 with the same distortion constraint.

Figure 4 illustrates the robustness against rotations. Wavelets drop to a detection rate of around 25% after a rotation of only $0.5°$ and it ceases to detect watermarks after a rotation of $1°$. On contrary, the robustness of our technique smoothly degrades with the strength of the attack.

Figure 5 illustrates the robustness against cropping. The performances again smoothly degrade with the strength of the attack: 50% of watermark detection when cropped to 90% of their original content and continues to detect some watermarks down to a crop of 50%. The system outperforms the classic approach. This is not surprising as DWT was not designed for geometric invariance. Yet, it shows that accurate synchronization is of utmost importance with linear transforms (*e.g.* DCT, DWT, random projection), whereas it is far less stringent in the new approach.

On top of this, we observe that the system is robust to horizontal flips! Since there are naturally and artificially occurring in the training dataset of ImageNet, the NN seems to have learned a transform invariant to this.

Figure 6 illustrates the robustness against JPEG compression. DWT performs better and we argue that this is caused by the lower dimensionality ($n = 4k$ *vs.* $1M$). The rule of thumb in watermarking is to spread the watermark to gain robustness against noise addition. This conflicts with the design of a domain invariant to geometric transformation.

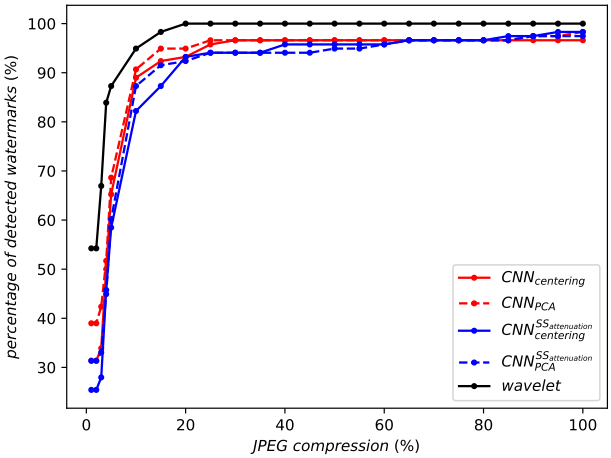

Figure 6. Robustness against JPEG compression.

6. Conclusion

This preliminary work shows that watermarking features extracted from a DL network is feasible and relevant. Feasible because the framework of adversarial sample provides a way to create watermarked image of good quality. Relevant because the DL network provides a transformation which strike a good trade-off between invariance to geometric attacks and robustness to valuemetric attacks.

However, it also raises some issues. First, the type of architecture of network has to be investigated. The Computer Vision literature keeps on improving the robustness of the image descriptor for image search [24]. Second, for a given architecture of the network, the training dataset plays also a big role. It is not clear at the moment to what extend invariance and robustness improve by learning on a training set dedicated to watermarking, *e.g.* comprising compressed, cropped, rotated images.

From the application point of view, this study opens the door to a single image descriptor good for both image search and watermarking. In copy detection and copyright infringement applications, image search alone yields many false positives. A watermark detection would drastically decrease the number of false recognition cases.

References

[1] Challenge on learned image compression. http://www.compression.cc/challenge/. Accessed: 2018-06-10. 4

[2] J. Alakuijala, R. Obryk, O. Stoliarchuk, Z. Szabadka, L. Vandevenne, and J. Wassenberg. Guetzli: Perceptually guided jpeg encoder. *arXiv preprint arXiv:1703.04421*, 2017. 4

[3] N. Carlini and D. Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 39–57, May 2017. 3

[4] P. Comesana, N. Merhav, and M. Barni. Asymptotically optimum universal watermark embedding and detection in the high-snr regime. *IEEE Transactions on Information Theory*, 56(6):2804–2815, 2010. 2

[5] I. Cox, M. Miller, and J. Bloom. *Digital Watermarking*. Morgan Kaufmann series in multimedia information and systems. Morgan Kaufmann, 2002. 2

[6] B. Darvish Rouhani, H. Chen, and F. Koushanfar. DeepSigns: A Generic Watermarking Framework for IP Protection of Deep Learning Models. *ArXiv e-prints*, Apr. 2018. 1

[7] D. A. E. Quiring and K. Rieck. Forgotten siblings: Unifying attacks on machine learning and digital watermarking. In *IEEE European Symposium on Security and Privacy*, 2018. 2

[8] T. Furon and P. Bas. Broken arrows. *EURASIP Journal on Information Security*, 2008(1):597040, Sep 2008. 2

[9] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and Harnessing Adversarial Examples. *ArXiv e-prints*, Dec. 2014. 3

[10] A. Gordo, J. Almazán, J. Revaud, and D. Larlus. Deep image retrieval: Learning global representations for image search. In *European Conference on Computer Vision*, pages 241–257. Springer, 2016. 1

[11] A. Iscen, G. Tolias, Y. Avrithis, T. Furon, and O. Chum. Panorama to panorama matching for location recognition. In *in Proceedings of ACM International Conference on Multimedia Retrieval (ICMR 2017)*, Bucharest, Romania, June 2017. ACM. 1

[12] C. Kanbak, S.-M. Moosavi-Dezfooli, and P. Frossard. Geometric robustness of deep networks: analysis and improvement. *arXiv preprint arXiv:1711.09115*, 2017. 1

[13] A. Khan, S. F. Tahir, A. Majid, and T.-S. Choi. Machine learning based adaptive watermark decoding in view of anticipated attack. *Pattern Recognition*, 41(8):2594 – 2610, 2008. 1

[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 1

[15] A. Kurakin, I. Goodfellow, and S. Bengio. Adversarial examples in the physical world. *ArXiv e-prints*, July 2016. 3

[16] N. Merhav and E. Sabbag. Optimal watermark embedding and detection strategies under limited detection resources. *IEEE Transactions on Information Theory*, 54(1):255–274, 2008. 2

[17] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard. Deepfool: A simple and accurate method to fool deep neural networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 3

[18] S. Mun, S. Nam, H. Jang, D. Kim, and H. Lee. A robust blind watermarking using convolutional neural network. *CoRR*, abs/1704.03248, 2017. 2

[19] Y. Nagai, Y. Uchida, S. Sakazawa, and S. Satoh. Digital watermarking for deep neural networks. *International Journal of Multimedia Information Retrieval*, 7(1):3–16, Mar 2018. 1

[20] F. Radenović, G. Tolias, and O. Chum. Cnn image retrieval learns from bow: Unsupervised fine-tuning with hard examples. In B. Leibe, J. Matas, N. Sebe, and M. Welling, editors, *Computer Vision – ECCV 2016*, pages 3–20, Cham, 2016. Springer International Publishing. 1

[21] O. Siméoni, A. Iscen, G. Tolias, Y. Avrithis, and O. Chum. Unsupervised object discovery for instance recognition. In *in Proceedings of Proceedings of IEEE Winter Conference on Applications of Computer Vision (WACV 2018)*, March 2018. 1

[22] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *ICLR*, 2015. 4

[23] J. Stolarek and P. Lipiński. Improving digital watermarking fidelity using fast neural network for adaptive wavelet synthesis. 2010. 1

[24] G. Tolias, R. Sicre, and H. Jégou. Particular object retrieval with integral max-pooling of cnn activations. *arXiv preprint arXiv:1511.05879*, 2015. 6

[25] J. Wu, Y. Yu, C. Huang, and K. Yu. Deep multiple instance learning for image classification and auto-annotation. In *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*, pages 3460–3469. IEEE, 2015. 1

[26] P.-T. Yu, H.-H. Tsai, and J.-S. Lin. Digital watermarking based on neural networks for color images. *Signal Processing*, 81(3):663 – 671, 2001. Special section on Digital Signal Processing for Multimedia. 1