WATERMARKING IMAGES IN SELF-SUPERVISED LATENT SPACES

Pierre Fernandez[1,2], *Alexandre Sablayrolles*[1], *Teddy Furon*[2], *Hervé Jégou*[1], *Matthijs Douze*[1]

[1]Meta AI, [2]Inria [*]

ABSTRACT

We revisit watermarking techniques based on pre-trained deep networks, in the light of self-supervised approaches. We present a way to embed both marks and binary messages into their latent spaces, leveraging data augmentation at marking time. Our method can operate at any resolution and creates watermarks robust to a broad range of transformations (rotations, crops, JPEG, contrast, etc). It significantly outperforms the previous zero-bit methods, and its performance on multi-bit watermarking is on par with state-of-the-art encoder-decoder architectures trained end-to-end for watermarking. The code is available at github.com/facebookresearch/ssl_watermarking

1. INTRODUCTION

Watermarking embeds a secret message into an image under the constraints of (1) *imperceptibility* – the distortion induced by the watermark must be invisible, (2) *payload* – the hidden message is a binary word of a given length, (3) *robustness* – the decoder retrieves the hidden message even if the image has been distorted to some extent. This paper deals with blind watermarking where the decoder does not have access to the original image. The classic approach, coined *TEmIt* (Transform, Embed, Inverse transform) by T. Kalker, embeds the watermark signal in the feature space of a transform (e.g. DFT, DCT, Wavelet). It provides *perceptually significant coefficients* reliable for watermarking as conceptualized in [2, Sec. 8.1.3].

Watermarking is enjoying renewed interest from advancements in deep learning. New methods improve the robustness to a broad range of alterations thanks to neural networks offering a reliable latent space where to embed the information. Examples include directly marking into the semantic space resulting from a supervised training over a given set of classes like ImageNet [3], or explicitly training a watermarking network to be invariant to a set of image perturbations. In this case, networks are usually encoder-decoder architectures trained end-to-end for watermarking [4, 5, 6, 7].

Our key insight is to leverage the properties of *self-supervised* networks to watermark images. Ideally, according to [2], a perceptually significant coefficient does not change unless the visual content of the image is different. Similarly, some self-supervised methods aim to create representations invariant to augmentations, without explicit knowledge of the image semantics [1, 8]. These pre-trained networks offer

us the desired embedding space "for free", saving us the heavy training of end-to-end architectures like HiDDeN [4].

In order to robustly embed in the latent spaces, gradient descent is performed over the pixels of the images. To further ensure both robustness and imperceptibility of the watermarks, we include data augmentation and image preprocessing at marking time.

Our contributions are the following:

- We provide a watermarking algorithm that can encode both marks and binary messages in the latent spaces of any pre-trained network;
- We leverage data augmentation at marking time;
- We experimentally show that networks trained with self-supervision provide excellent embedding spaces.

2. RELATED WORK

Image watermarking [2] approaches are often classified by their embedding space: few use the spatial domain [9, 10], most of them follow the TEmIt principle with a well known transformation like DFT [11], DCT [12] or DWT [13]. In **zero-bit watermarking** [14], the embedder only hides a mark and the *detector* checks for its presence in the content. For instance, the received content is deemed watermarked if its descriptor lies in a hypercone of the feature space. This detection strategy is near optimal [15, 16]. In **multi-bit watermarking**, the embedder encodes a binary word into a signal that is hidden in the image. At the other end, the *decoder* retrieves the hidden message bit by bit. This is the case for most deep-learning-based methods presented below.

Deep-learning-based watermarking has emerged as a viable alternative to traditional methods. The networks are often built as encoder-decoder architectures [4, 17], where an encoder embeds a message in the image and a decoder tries to extract it. For instance, HiDDeN [4] jointly trains encoder and decoder networks with noise layers that simulate image perturbations. It uses an adversarial discriminator to improve visual quality, and was extended to arbitrary image resolutions and message lengths by Lee et al. [18]. Distortion Agnostic [19] adds adversarial training that brings robustness to unknown transformations and [7, 20] embed the mark with an attention filter further improving imperceptibility. ReDMark [6] adds a circular convolutional layer that diffuses the watermark signal all over the image. Finally, ROMark [5] uses robust optimization with worst-case attack as if an adversary were trying to remove the mark. For more details, we refer to the review [21]. This type of methods has its drawbacks: e.g. it is hard to control the embedding dis-
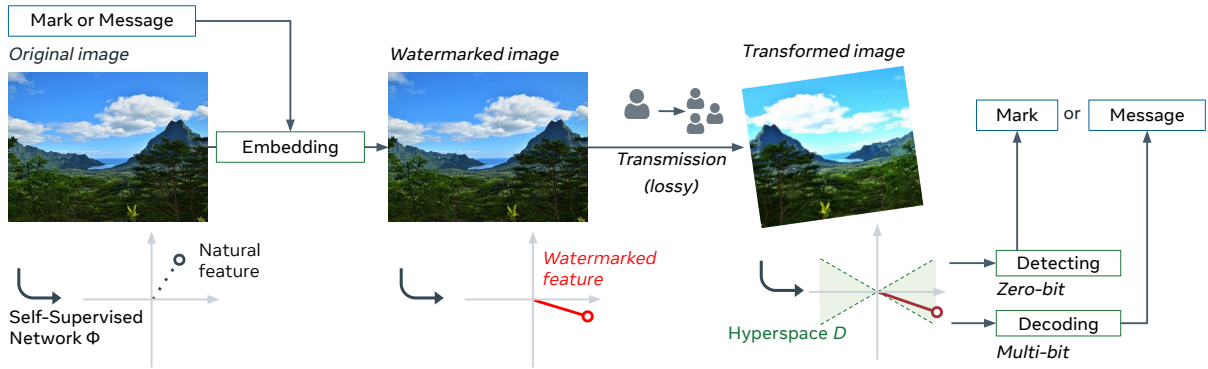
Fig. 1: Overview of our method for watermarking in self-supervised-latent spaces. A self-supervised network trained with DINO [1] builds a latent space on which the mark is added by the embedding process. Its effect is to shift the image's feature into a well-specified region of the latent space, such that transformations applied during transmission do not move much the feature. The mark's detection (zero-bit watermarking setup) or message's decoding (multi-bit watermarking setup) is performed in the same latent space.

tortion and it is made for decoding and does not translate so well to detection.

Vukotić et al. [22] mark images with a neural network pre-trained on supervised classification instead of an encoder-decoder architecture. The network plays the role of the transform in a TEmIt approach. Since it has no explicit inverse, a gradient descent to the image pixels "pushes" the image feature vector into a hypercone. The follow-up work [3] increases the inherent robustness of the network by applying increasingly harder data augmentation at pre-training. It offers a guarantee on the false positive rate without requiring to train a network explicitly for watermarking, but no multi-bit version was proposed.

3. WATERMARKING WITH SSL NETWORKS

Our method adopts the framework of Vukotić et al. [3]. We improve it by showing that networks build better watermarking features when trained with self-supervision, and by introducing data-augmentation at marking time. We also extend it to multi-bit watermarking.

3.1. Using self-supervised networks as feature extractors

Motivation. We denote the image space by $\mathcal{I}$ and the feature space by $\mathcal{F} = \mathbb{R}^d$. The feature extractor $\phi : \mathcal{I} \rightarrow \mathcal{F}$ must satisfy two properties: (1) geometric and valuemetric transformations on image I should have minimal impact on feature $\phi(I)$, (2) it should be possible to alter $\phi(I)$ by applying invisible perturbations to the image in order to embed the watermark signal.

We choose ϕ as a neural network pre-trained by self-supervised learning (SSL). Our assumption is that SSL produces excellent marking spaces because (1) it explicitly trains features to be invariant to data augmentation; and (2) it does not suffer from the *semantic collapse* of supervised classification, that gets rid off any information that is not necessary to assign classes [23]. From the recent SSL methods of the literature (contrastive learning [24, 25], statistical prior [26], teacher/ student architecture [8]), we select DINO [1] for its training speed and content-based image retrieval performance.

DINO pre-training. DINO [1] pertains to the teacher/ student approach. The teacher and the student share the same architecture. Self distillation with no labels trains the student network to match the outputs of the teacher on different views of the same image. The student is updated by gradient descent while the teacher's parameters are updated as an exponential moving average of the student's parameters: $\theta_t \leftarrow \lambda \theta_t + (1 - \lambda)\theta_s$, with $\lambda \lesssim 1$.

The invariance of the features is ensured by random augmentations during the training: valuemetric (color jittering, Gaussian blur, solarization) and geometric (resized crops) transformations. Furthermore, DINO encourages local-to-global correspondence by feeding only global views to the teacher while the student sees smaller crops.

Normalization layer. The watermark embedding must drive the feature to an arbitrary space region (defined by a secret key and the message to hide). It is essential that the features are not concentrated onto a manifold far away from this region. Although Wang et al. [27] show that contrastive learning optimizes the uniformity of the features on the hypersphere, it does not occur in practice with DINO. To alleviate the issue, the output features are transformed by PCA-whitening (a.k.a. PCA-sphering). This learned linear transformation [28] outputs centered vectors with unit covariance of dimension $d = 2048$.

3.2. Marking with back-propagation and augmentation

The marking takes an original image I_o and outputs a visually similar image I_w. In the image space $\mathcal{I}$, the distortion is measured by $\mathcal{L}_i : \mathcal{I} \times \mathcal{I} \rightarrow \mathbb{R}_+$. An example is the MSE: $\mathcal{L}_i(I_w, I_o) = \|I_w - I_o\|^2/h/w$, but it could be replaced by perceptual image losses such as LPIPS [29].

In the feature space $\mathcal{F}$, we define a region $\mathcal{D}$ which depends on a secret key (zero-bit and multi-bit setups) and the message to be hidden (only in multi-bit setup). Its definition is deferred to Sect. 3.3 together with the loss $\mathcal{L}_w : \mathcal{F} \rightarrow \mathbb{R}$ that captures how far away a feature $x \in \mathcal{F}$ lies from $\mathcal{D}$. We also define a set $\mathcal{T}$ of augmentations, which include rotation, crops, blur, etc., each with a range of parameters. $\mathrm{Tr}(I, t) \in \mathcal{I}$ denotes the application of transformation $t \in \mathcal{T}$ to image I.

If the feature extractor is perfectly invariant, $\phi(\mathrm{Tr}(I, t))$

lies inside $\mathcal{D}$ if $\phi(I)$ does. To ensure this, the watermarking uses data augmentation. The losses $\mathcal{L}_w$ and $\mathcal{L}_i$ are combined in:

$$\mathcal{L}(I, I_o, t) := \lambda \mathcal{L}_w(\phi(\mathrm{Tr}(I, t))) + \mathcal{L}_i(I, I_o). \quad (1)$$

The term $\mathcal{L}_w$ aims to push the feature of *any transformation* of I_w into $\mathcal{D}$, while the term $\mathcal{L}_i$ favors low distortion. The training approach is typical for the adversarial attacks literature [30, 31]:

$$I_w := \arg \min_{I \in \mathcal{C}(I_o)} \mathbb{E}_{t \sim \mathcal{T}}[\mathcal{L}(I, I_o, t)] \quad (2)$$

where $\mathcal{C}(I_o) \subset \mathcal{I}$ is the set of admissible images w.r.t. the original one. It is defined by two steps of normalization applied to the pixel-wise difference $\delta = I - I_o$: (1) we apply a SSIM [32] heatmap attenuation, which scales δ pixel-wise to hide the information in perceptually less visible areas of the image; (2) we set a minimum target PSNR and rescale δ if this target is exceeded.

The minimization is performed by stochastic gradient descent since the quality constraints, Tr and ϕ are differentiable w.r.t. the pixel values. Stochasticity comes from the fact that expectation $\mathbb{E}_{t \sim \mathcal{T}}$ is approximated by sampling according to a distribution over $\mathcal{T}$. The final image is the rounded version of the update after K iterations.

3.3. Detection and decoding

We consider two scenarios: zero-bit (detection only) and multi-bit watermarking (decoding the hidden message). Contrary to HiDDeN [4] and followers, our decoding is mathematically sound.

Zero-bit. From a secret key $a \in \mathcal{F}$ s.t. $\|a\| = 1$, the detection region is the dual hypercone:

$$\mathcal{D} := \left\{ x \in \mathbb{R}^d : |x^\mathsf{T} a| > \|x\| \cos(\theta) \right\}. \quad (3)$$

It is well grounded because the False Positive Rate (FPR) is given by

$$\mathrm{FPR} := \mathbb{P}\left(\phi(I) \in \mathcal{D} \mid \text{"key } a \text{ uniformly distributed"}\right)$$
$$= 1 - I_{\cos^2(\theta)}\left(\frac{1}{2}, \frac{d-1}{2}\right) \quad (4)$$

where $I_\tau(\alpha, \beta)$ is the regularized Beta incomplete function. Moreover, the best embedding is obtained by increasing the following function under the distortion constraint:

$$-\mathcal{L}_w(x) = (x^\mathsf{T} a)^2 - \|x\|^2 \cos^2 \theta. \quad (5)$$

This quantity is negative when $x \notin \mathcal{D}$ and positive otherwise. I. Cox et al. originally called it the robustness estimate [2, Sec. 5.1.3]. This hypercone detector is optimal under the asymptotical Gaussian setup in [15, 16].

Multi-bit. We now assume that the message to be hidden is $m = (m_1, ..., m_k) \in \{-1, 1\}^k$. The decoder retrieves $\hat{m} = D(I)$. Here, the secret key is a randomly sampled orthogonal family of carriers $a_1, ..., a_k \in \mathbb{R}^d$. We modulate m into the signs of the projection of the feature $\phi(I)$ against each of the carriers, so the decoder is:

$$D(I) = \left[\mathrm{sign}\left(\phi(I)^\top a_1\right), ..., \mathrm{sign}\left(\phi(I)^\top a_k\right) \right]. \quad (6)$$

At marking time, the functional is now defined as the hinge loss with margin $\mu \geq 0$ on the projections:

$$\mathcal{L}_w(x) = \frac{1}{k} \sum_{i=1}^{k} \max\left(0, \mu - (x^\top a_i).m_i\right). \quad (7)$$

3.4. Overview of the watermarking algorithm

Algorithm 1 Watermarking algorithm

Inputs: $I_0 \in \mathcal{I}$, targeted PSNR in dB
  if zero-bit: FPR $\in [0, 1]$, $a \in \mathbb{R}^d$
  if multi-bit: $m \in \{-1, 1\}^k$, $(a_i)_{i=1...k} \in \mathbb{R}^{k \times d}$
Embedding:
  if zero-bit: compute $\theta(\mathrm{FPR})$
  $I_w \leftarrow I_0$
  For $i = 1, ..., n_{iter}$:
    $I_w \xleftarrow{\text{constraints}} I_w$ $\triangleright$ impose constraints
    $I_w \xleftarrow{t \sim \mathcal{T}} \mathrm{Tr}(I_w, t)$ $\triangleright$ sample & apply augmentation
    $x \leftarrow \phi(I_w)$ $\triangleright$ get feature of the image
    $\mathcal{L} \leftarrow \lambda \mathcal{L}_w(x) + \mathcal{L}_i(I_w, I_0)$ $\triangleright$ compute loss
    $I_w \leftarrow I_w + \eta \times \mathrm{Adam}(\mathcal{L})$ $\triangleright$ update the image
  Return I_w
Detecting:
  $x \leftarrow \phi(I_m)$
  if zero-bit:
    Detect if $x \in \mathcal{D} \iff (x^T a)^2 - \|x\|^2 \cos^2 \theta > 0$
  if multi-bit:
    Return $\left[\mathrm{sign}\left(x^\top a_1\right), ..., \mathrm{sign}\left(x^\top a_k\right) \right]$
---

4. EXPERIMENTS & RESULTS

4.1. Experimental setup and implementation details

Data. We evaluate our method on: 1000 images of YFCC100M dataset [33] which is selected for the variety of its content, CLIC [34] composed of 118 professional high-resolution images when comparing to [3], and 1000 images of MS-COCO [35] composed of smaller images for comparison with [4, 19].

Backbone pre-training. We use the ResNet-50 [36] architecture as backbone model to extract features from its last convolutional layer ($d = 2048$). It is trained on ILSVRC2012 [37] without labels, using 200 epochs of DINO self-supervised learning with the default parameters [1] and with rotation augmentation. Models trained for classification come from the torchvision library [38]. The PCA-whitening is learned on 100k distinct images from YFCC (resp. COCO) when evaluating on YFCC and CLIC (resp. COCO).

Embedding. We first set a desired FPR (which defines the hypercone angle θ) and a target PSNR. Watermarking (2) then uses the Adam optimizer [39] with learning rate 0.01 over 100 gradient descent iterations. The weight in (1) is set to $\lambda = 1$ (zero-bit) or $\lambda = 5 \cdot 10^4$ (multi-bit). The margin of (7) is set to $\mu = 5$.

At each iteration, the preprocessing step performs the SSIM attenuation and clips the PSNR to the target. SSIM heatmaps are computed with $C_1 = 0.01^2$, $C_2 = 0.03^2$ and over 17×17 tiles of the image's channels, then summed-up and clamped to be non negative, which generates a single heatmap per image. Then, a transformation t is chosen randomly in $\mathcal{T}$ (identity, rotation, blur, crop or resize). The rotation angle α is sampled from a Von Mises distribution with $\mu = 0$, $\kappa = 1$ and divided by 2. This generates angles in $\pi/2 \times [-1, 1]$ with a higher probability for small rotations, that are more frequent in practice. The crop and resize scales are chosen uniformly in $[0.2, 1.0]$. The crop aspect ratio is also randomly chosen between $3/4$ and $4/3$. The blurring kernel size b is randomly drawn from the odd numbers between 1 and 15 and σ is set to $0.15b + 0.35$. Finally, the image is flipped horizontally with probability 0.5.

Transformations. The following table presents the transformations used at pre-training, marking or evaluation stages. Parameter p represents the cropping ratio in terms of area, Q is the quality factor of the compression and B, C, H are defined in [38]. "Meme format" and "Phone screenshot" come from the Augly library [40].

Transformations	Parameter	Type		Used for	
		Geom	Value	Train	Mark
Rotation	angle α	✓	✗	✓	✓
Crop	ratio p	✓	✗	✓	✓
Resize	scale p	✓	✗	✓	✓
Gaussian Blur	width σ	✗	✓	✓	✓
Brightness	B	✗	✓	✓	✗
Contrast	C	✗	✓	✓	✗
Hue	H	✗	✓	✓	✗
JPEG	quality Q	✗	✓	✗	✗
Meme format	-	✓	✓	✗	✗
Phone screenshot	-	✓	✓	✗	✗

4.2. Zero-bit watermarking

Trade-offs. The hypercone angle θ in (4) is given by the target FPR. A higher FPR implies a wider angle, making the method robust against more severe attacks, at the cost of detecting more false positives. The FPR is set to 10^{-6} in further experiments. Large-scale applications usually operate at low FPR to avoid human verification. As a sanity check we run detection on 100k natural images from YFCC, none of which are found to be marked. Similarly, there is only one false positive out of the 1,281,167 images of ImageNet. Another trade-off lies in the imperceptibility, since allowing greater distortions (lower PSNR) improves the robustness. It is illustrated in Fig. 2.

Ablation studies. We showcase the influence of self supervision at pre-training and of augmentation at marking time. The performance measure is the True Positive Rate (TPR), at a target PSNR=40 dB, and FPR=10^{-6}. Fig. 3 evaluates the robustness for the specific case of the rotation. Rotation augmentation is needed both at pre-training and marking stages to achieve high robustness against it. Comparison on a wider range of transformations is given in Tab. 1.

Qualitative results. Fig. 4 presents an image watermarked at PSNR=40 dB and some detected alterations, as well as its pixel-wise difference w.r.t. the original image. The watermark is almost invisible even to the trained eye because it is added in the textured regions due to the perceptual SSIM normalization applied during watermarking.
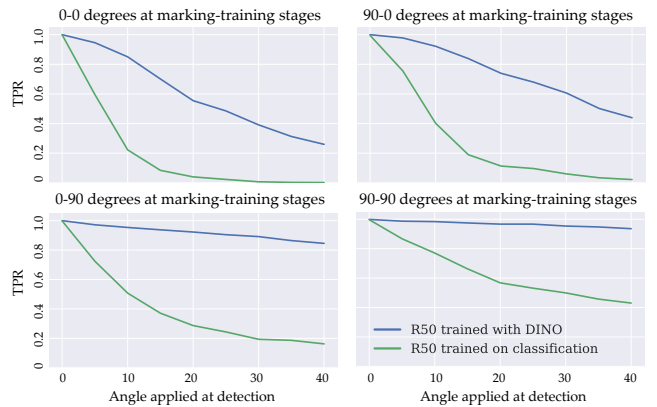

Fig. 3: Robustness against rotation. Each row (column) represents different ±amplitude of the rotation at training (resp. marking (3)).
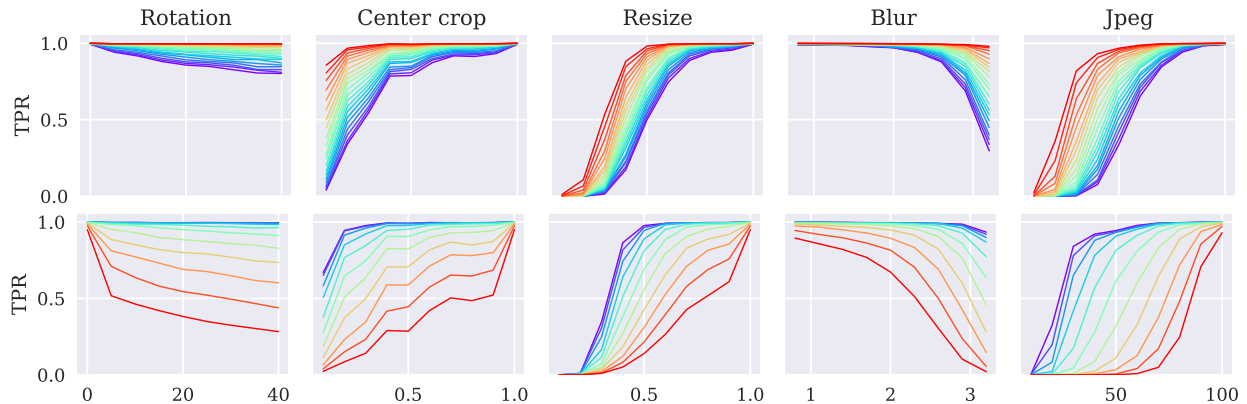

Fig. 2: Robustness of the detection in the zero-bit setup against image transformations. *Top:* PSNR set at 40 dB and FPR decreasing from 10^{-2} (red) to 10^{-12} (blue). *Bottom:* FPR set at 10^{-6} and PSNR ranging from 52 dB to 32 dB.

Table 1: TPR over various attacks. 1st setup: performance with SSL vs supervised ResNet-50 networks on YFCC. 2nd setup: evaluation over CLIC. (★) best results in [3] (VGG-19 trained on hard attacks and RMAC aggregation), (★★) our implementation of [3] (with default pre-trained VGG-19).
† denotes augmentations used at pre-training.

Transformations	Setup 1: YFCC		Setup 2: CLIC		
	SSL	Sup.	Ours	[3] (★)	[3] (★★)
Identity	$1.00^{\dagger}$	$1.00^{\dagger}$	$1.00^{\dagger}$	$1.0^{\dagger}$	$1.00^{\dagger}$
Rotation (25)	$0.97^{\dagger}$	$0.54^{\dagger}$	$\mathbf{1.00}^{\dagger}$	$\approx 0.3^{\dagger}$	$0.27^{\dagger}$
Crop (0.5)	$0.95^{\dagger}$	$0.79^{\dagger}$	$\mathbf{1.00}^{\dagger}$	$\approx 0.1^{\dagger}$	$1.00^{\dagger}$
Crop (0.1)	$0.39^{\dagger}$	$0.06^{\dagger}$	$\mathbf{0.98}^{\dagger}$	$\approx 0.0^{\dagger}$	$0.02^{\dagger}$
Resize (0.7)	$0.99^{\dagger}$	$0.85^{\dagger}$	$1.00^{\dagger}$	-	$1.00^{\dagger}$
Blur (2.0)	$0.99^{\dagger}$	0.04	$\mathbf{1.00}^{\dagger}$	-	0.25
JPEG (50)	0.81	0.20	0.97	≈ 1.0	0.96
Brightness (2.0)	$0.94^{\dagger}$	0.71	$0.96^{\dagger}$	-	0.99
Contrast (2.0)	$0.96^{\dagger}$	0.65	$1.00^{\dagger}$	-	1.00
Hue (0.25)	$1.00^{\dagger}$	0.46	$1.00^{\dagger}$	-	1.00
Meme	0.99	0.94	1.00	-	0.98
Screenshot	0.76	0.18	$\mathbf{0.97}$	-	0.86

Table 2: BER and WER (%) for 30-bits encoding at PSNR 40dB.

Transform.	Identity	Rot. (25)	Crop (0.5)	Crop (0.1)	Res. (0.7)	Blur (2.0)	JPEG (50)	Bright. (2.0)	Contr. (2.0)	Hue (0.25)	Meme	Screenshot
BER	0.1	3.3	4.8	28.9	2.1	0.5	20.8	8.7	8.4	2.5	6.4	23.9
WER	0.7	43.4	58.3	100	29.1	4.6	98.9	60.7	62.6	31.6	75.9	100

Table 3: Comparison of BER. The first row uses original resolutions of COCO, while the others use a resized version (to 128×128). Results for [4, 19] come from [19]. † denotes transformations used in the watermarking process.

Transformation	Identity	JPEG (50)	Blur (1.0)	Crop (0.1)	Resize (0.7)	Hue (0.2)
Ours	$0.00^{\dagger}$	0.04	$0.00^{\dagger}$	$0.18^{\dagger}$	$0.00^{\dagger}$	0.03
Ours, 128×128	$0.00^{\dagger}$	0.16	$0.01^{\dagger}$	$0.45^{\dagger}$	$0.18^{\dagger}$	0.06
HiDDeN [4]	$0.00^{\dagger}$	$0.23^{\dagger}$	$0.01^{\dagger}$	$0.00^{\dagger}$	0.15	0.29
Dist. Agnostic [19]	$0.00^{\dagger}$	$0.18^{\dagger}$	$0.07^{\dagger}$	$0.02^{\dagger}$	0.12	0.06

Comparison with the state of the art. Table 1 compares our method with [3] on CLIC. In their setup, the FPR=10^{-3} and PSNR must be ≥ 42dB. Overall, our method gives better results on CLIC than on YFCC because images have higher resolutions (hence more pixels can be used to convey the mark). We observe a strong improvement over [3], especially for large rotations, crops and Gaussian blur where our method yields almost perfect detection over the 118 images.

4.3. Multi-bit data hiding

Quantitative results. We evaluate the method on YFCC, with a target PSNR of 40 dB and a payload k of 30 random bits as in [4, 19]. Tab. 2 presents the Bit and Word Error Rate (BER and WER) over various attacks. The decoding achieves low rates over a wide range of geometric (rotation, crops, resize, etc.) and valuemetric (brightness, hue, contrast, etc.) attacks. Rotation and Gaussian blur are particularly harmless since they are seen both at pre-training and at marking time. Some images are harder to mark, which can be observed statistically on the empiricial WER reported in Tab. 2. If all images were as difficult to mark, then BER would be equal for all images, and WER $= 1 - (1 - \text{BER})^k$. Yet, the reported WER are significantly lower: e.g. for Brigthness, WER $= 0.607 < 1 - (1 - 0.087)^{30} = 0.935$. Empirically, we

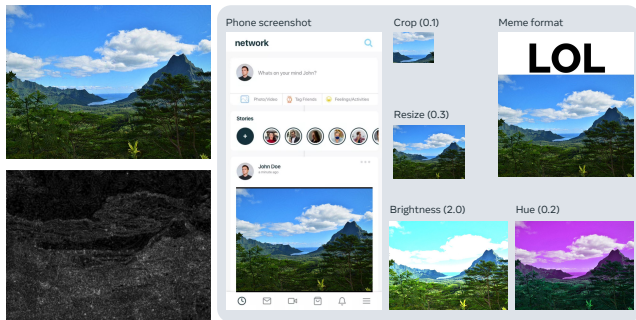see that images with little texture are harder to watermark than others, due to the SSIM normalization. In practice, ECC can be used to achieve lower WERs.

Qualitative results. We notice that the watermark is perceptually less visible for multi-bit than for zero-bit watermarking at a fixed PSNR. Our explanation is that the energy put into the image feature is more spread-out across carriers in the multi-bit setup than on the zero-bit one where the feature is pushed at much as possible towards a single carrier. Images are not displayed due to lack of space.

Comparison with the state of the art. Table 3 compares against two deep data hiding methods [4, 19] using their setting: a payload of 30 bits, a target PSNR of 33dB, over 1000 images from COCO resized to 128×128. Overall, our method gives comparable results to the state of the art, except for the center crop transform where it fails to achieve high bit accuracies. Note also that the resize operation is not used as noise layer in neither of [4, 19], which means that our method should have the advantage. In contrast, while these methods are trained to be robust to JPEG compression with a differentiable approximation, our method achieves similar performance without specific training.

Furthermore, our method easily scales to higher resolution images, where it achieves lower BER for a fixed payload. We assume that [4, 19] also scales but at the cost of a specific training for a given resolution. This training is more computationally expensive since the message is repeated at each pixel [4]. It also needs a smaller batch-size to operate larger images, and new hyperparameters values.

5. CONCLUSION & DISCUSSION

This paper proposes a way to robustly and invisibly embed information into digital images, by watermarking onto latent spaces of off-the-shelf self-supervised networks. By incorporating data augmentation and constraints into the marking process, our zero-bit watermarking method greatly improves performance over the baseline [3]. It is robust against

Fig. 4: Example of an image (800×600) watermarked at PSNR 40 dB and FPR=10^{-6}, and some detected alterations. The black and white picture shows the scaled amplitude of the watermark signal.

a wide range of transformations while keeping high fidelity with regards to the original images, and ensuring a very low false positive rate for the detection. When we extend the method to multi-bit watermarking, we obtain promising results, comparable to the state-of-the-art in deep data hiding, and even better with regards to some transformations of the image (e.g. JPEG compression or blur).

Most interestingly, networks trained with self-supervision naturally generate excellent watermarking spaces, without being explicitly trained to do so. However, compared to encoder-decoder deep watermarking techniques, watermarking images with our method is expansive since it is not a single pass forward. In future works, we hope to show that further adapting the network for the specific task of watermarking would improve performance and efficiency.

6. REFERENCES

[1] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin, "Emerging properties in self-supervised vision transformers," *ICCV*, 2021.

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

[3] Vedran Vukotić, Vivien Chappelier, and Teddy Furon, "Are classification deep neural networks good for blind image watermarking?," *Entropy*, 2020.

[4] Jiren Zhu, Russell Kaplan, Justin Johnson, and Li Fei-Fei, "Hidden: Hiding data with deep networks," in *ECCV*, 2018.

[5] Bingyang Wen and Sergul Aydore, "Romark: A robust watermarking system using adversarial training," *arXiv preprint arXiv:1910.01221*, 2019.

[6] Mahdi Ahmadi, Alireza Norouzi, Nader Karimi, Shadrokh Samavi, and Ali Emami, "Redmark: Framework for residual diffusion watermarking based on deep networks," *Expert Systems with Applications*, 2020.

[7] Honglei Zhang, Hu Wang, Yuanzhouhan Cao, Chunhua Shen, and Yidong Li, "Robust watermarking using inverse gradient attention," *arXiv preprint arXiv:2011.10850*, 2020.

[8] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, et al., "Bootstrap your own latent: A new approach to self-supervised learning," *NeurIPS*, 2020.

[9] Zhicheng Ni, Yun-Qing Shi, Nirwan Ansari, and Wei Su, "Reversible data hiding," *IEEE Transactions on circuits and systems for video technology*, 2006.

[10] Nikos Nikolaidis and Ioannis Pitas, "Robust image watermarking in the spatial domain," *Signal processing*, 1998.

[11] Matthieu Urvoy, Dalila Goudia, and Florent Autrusseau, "Perceptual dft watermarking with improved detection and robustness to geometrical distortions," *IEEE Transactions on Information Forensics and Security*, 2014.

[12] Adrian G Bors and Ioannis Pitas, "Image watermarking using dct domain constraints," in *ICIP*, 1996.

[13] Xiang-Gen Xia, Charles G Boncelet, and Gonzalo R Arce, "Wavelet transform based watermark for digital images," *Optics Express*, 1998.

[14] Teddy Furon, "A constructive and unifying framework for zero-bit watermarking," *IEEE Transactions on Information Forensics and Security*, 2007.

[15] Teddy Furon, "Watermarking error exponents in the presence of noise: The case of the dual hypercone detector," in *ACM Workshop on Information Hiding and Multimedia Security*, 2019.

[16] Neri Merhav and Erez Sabbag, "Optimal watermark embedding and detection strategies under limited detection resources," *IEEE Transactions on Information Theory*, 2008.

[17] Haribabu Kandi, Deepak Mishra, and Subrahmanyam R.K. Sai Gorthi, "Exploring the learning capabilities of convolutional neural networks for robust image watermarking," *Computers & Security*, 2017.

[18] Jae-Eun Lee, Young-Ho Seo, and Dong-Wook Kim, "Convolutional neural network-based digital image watermarking adaptive to the resolution of image and watermark," *Applied Sciences*, 2020.

[19] Xiyang Luo, Ruohan Zhan, Huiwen Chang, Feng Yang, and Peyman Milanfar, "Distortion agnostic deep watermarking," in *CVPR*, 2020.

[20] Chong Yu, "Attention based data hiding with generative adversarial networks," in *AAAI*, 2020.

[21] Olivia Byrnes, Wendy La, Hu Wang, Congbo Ma, Minhui Xue, and Qi Wu, "Data hiding with deep learning: A survey unifying digital watermarking and steganography," *arXiv preprint arXiv:2107.09287*, 2021.

[22] Vedran Vukotić, Vivien Chappelier, and Teddy Furon, "Are deep neural networks good for blind image watermarking?," in *WIFS*, 2018.

[23] Carl Doersch, Ankush Gupta, and Andrew Zisserman, "Crosstransformers: spatially-aware few-shot transfer," *NeurIPS*, 2020.

[24] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton, "A simple framework for contrastive learning of visual representations," in *ICML*, 2020.

[25] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick, "Momentum contrast for unsupervised visual representation learning," in *CVPR*, 2020.

[26] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny, "Barlow twins: Self-supervised learning via redundancy reduction," *ICML*, 2021.

[27] Tongzhou Wang and Phillip Isola, "Understanding contrastive representation learning through alignment and uniformity on the hypersphere," in *ICML*, 2020.

[28] Hervé Jégou and Ondřej Chum, "Negative evidences and co-occurences in image retrieval: The benefit of pca and whitening," in *ECCV*, 2012.

[29] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *CVPR*, 2018.

[30] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy, "Explaining and harnessing adversarial examples," *ICLR*, 2014.

[31] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus, "Intriguing properties of neural networks," *ICLR*, 2013.

[32] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on image processing*, 2004.

[33] Bart Thomee, David A. Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li, "Yfcc100m: The new data in multimedia research," *Communications of the ACM*, 2016.

[34] "Workshop and challenge on learned image compression," .

[35] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick, "Microsoft coco: Common objects in context," in *ECCV*, 2014.

[36] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *CVPR*, 2016.

[37] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *CVPR*, 2009.

[38] Sébastien Marcel and Yann Rodriguez, "Torchvision the machine-vision package of torch," in *ACM International Conference on Multimedia*, 2010.

[39] Diederik P. Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," in *ICLR*, 2015.

[40] Joanna Bitton and Zoe Papakipos, "Augly: A data augmentations library for audio, image, text, and video," 2021.

Appendix

6.1. Demonstration of the False Positive Rate formula (Eq. 4)

One way to sample U is the following: First sample a white Gaussian vector $G \sim \mathcal{N}(0, I)$ in $\mathbb{R}^d$ and then normalize: $U = G/\|G\|$. Without loss of generality, we can assume that $a = (1, 0, ..., 0)$ (up to a change of axes). Therefore,

$$U^T a = \frac{G_1}{\sqrt{\sum_{i=1}^{d} G_i^2}}$$

Let $\tau \in [0, 1]$. We have

$$\mathbb{P}\left((U^T a)^2 \geq \tau^2\right) = \mathbb{P}\left(\frac{G_1^2}{\sum_{i=1}^{d} G_i^2} > \tau^2\right) = \mathbb{P}\left(\frac{G_1^2}{\sum_{i=2}^{d} G_i^2} > \frac{\tau^2}{1 - \tau^2}\right)$$

$$= \mathbb{P}\left((d-1)\frac{G_1^2}{\sum_{i=2}^{d} G_i^2} > (d-1)\frac{\tau^2}{1 - \tau^2}\right)$$

Note that $Y := (d-1)\frac{G_1^2}{\sum_{i=2}^{d} G_i^2}$ is the ratio of two independent chi-squares random variables of degree 1 and $d-1$. By definition [?], Y follows a Fisher distribution $F(1, d-1)$. Its cumulative density function is: $F(x; 1, d-1) = I_{x/(x+d-1)}(1/2, (d-1)/2)$.

It follows, using $x = (d-1)\frac{\tau^2}{1-\tau^2} = -(d-1) + (d-1)\frac{1}{1-\tau^2}$ and setting $\tau = \cos\theta$, that

$$\mathbb{P}(|U^T a| > \cos\theta) = \mathbb{P}(Y > \cos^2\theta) = 1 - I_{\cos^2\theta}(1/2, (d-1)/2) = I_{\sin^2\theta}((d-1)/2, 1/2)$$
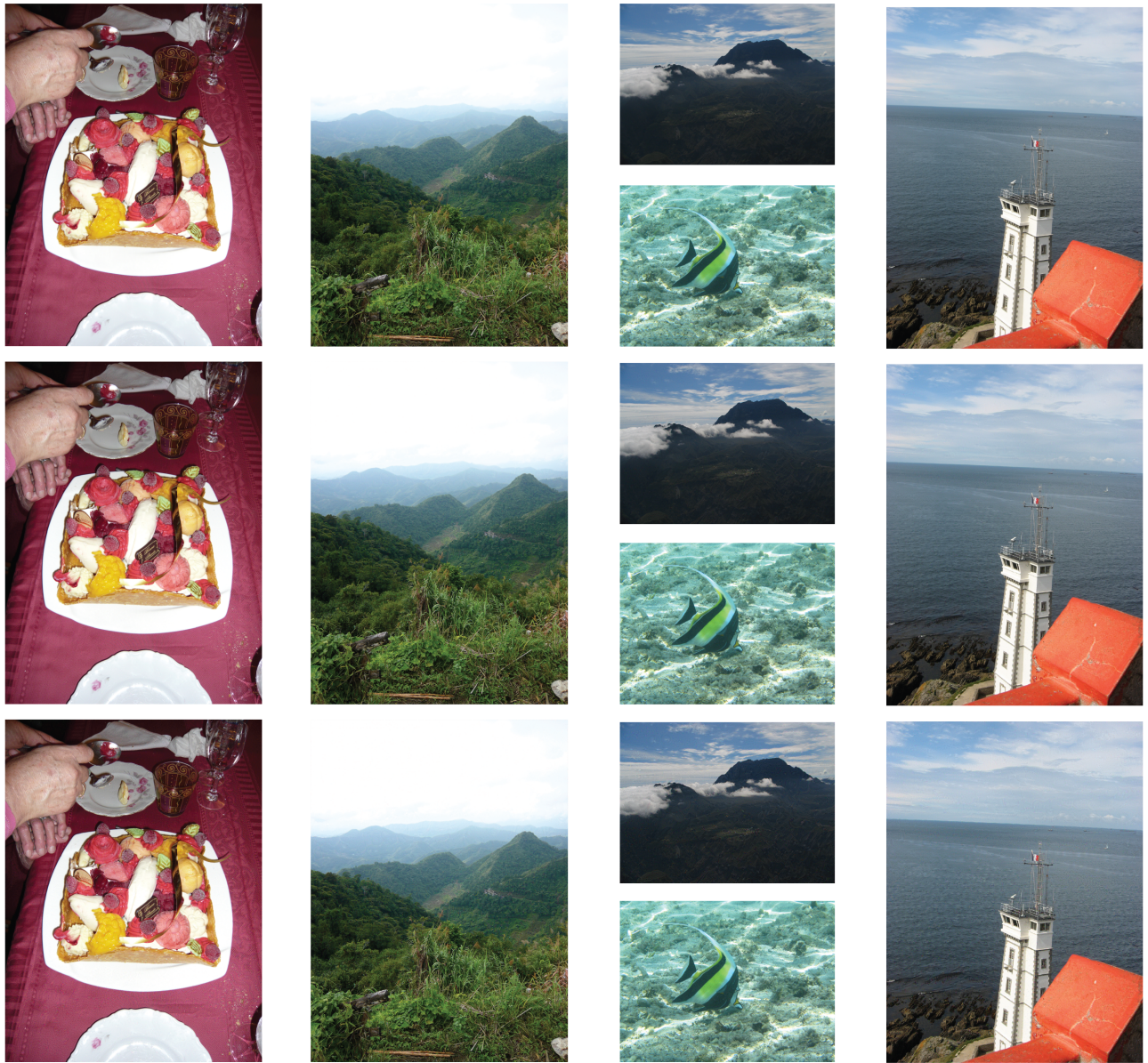
where $I_x(a, b)$ is the regularized incomplete beta function.

6.2. Low resolution watermarked images

Fig. 5: Watermarked images from the INRIA Holidays dataset resized to 128×128. The watermark is added with our multi-bit watermarking method with a payload of 30 bits and with different values for the target PSNR: 52dB (top row), 40dB (middle row), 32dB (bottom row). We use a 5-bits character encoding to encode and decode one message per image, and show the decoded messages for each image (without any transformation applied to the image). The higher the PSNR, the higher the decoding errors, and the less robust the decoding is to transformations.

6.3. High resolution watermarked images

(a) The watermark is added with our zero-bit watermarking method with an FPR of 10^{-6} and with different values for the target PSNR: 52dB (top row), 40dB (middle row), 32dB (bottom row)

(b) The watermark is added with our multi-bit watermarking method with a payload of 30 bits with the target PSNR at 32dB.

Fig. 6: Watermarked images from the INRIA Holidays dataset (resolution around 2048×1536).