Tatouage Numérique d'Images dans l'Espace Latent de Réseaux Auto-Supervisés

Pierre Fernandez, Alexandre Sablayrolles, Teddy Furon, Hervé Jégou, Matthijs Douze

Tatouage Numérique d'Images dans l'Espace Latent de Réseaux Auto-Supervisés

Pierre FERNANDEZ[1,2], Alexandre SABLAYROLLES[1], Teddy FURON[2], Hervé JÉGOU[1], Matthijs DOUZE[1],

[1]Meta AI, FAIR
6 rue Menars, 75002 Paris, France

[2]Univ. Rennes, Inria, CNRS, IRISA
Campus de Beaulieu, 263 avenue du Général Leclerc, 35042 Rennes cedex, France

Correspondance: Pierre Fernandez <pfz@fb.com>

Résumé – Nous revisitons les techniques de tatouage numérique basées sur des transformations d'images, à la lumière des réseaux neuronaux auto-supervisées. Nous présentons un moyen d'intégrer à la fois des marques et des messages binaires dans leurs espaces latents. Notre méthode peut fonctionner à résolution variable et crée des tatouages robustes à un large éventail de transformations (rotations, recadrages, JPEG, contraste, etc.). Elle surpasse de manière significative les précédentes méthodes de tatouage zéro-bit, et ses performances en multi-bits sont comparables à celles des architectures encodeur-décodeur entraînées de bout en bout pour le tatouage numérique. Le code est disponible sur github.com/facebookresearch/ssl_watermarking.

Abstract – We revisit watermarking techniques based on image transformations, in the light of self-supervised neural networks. 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, Recadrages, JPEG, Contraste, 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

Le tatouage numérique intègre une marque dans une image sous les contraintes suivantes (1) *imperceptibilité* - la distorsion induite par le tatouage doit être invisible, (2) *robustesse* - le message caché peut être retrouvé même si l'image a été déformée dans une certaine mesure, (3) *sécurité* - le message est secret. Cet article traite du tatouage aveugle où le décodeur n'a pas accès à l'image originale.

L'approche classique, nommée *TEmIt* (Transform, Embed, Inverse transform) par T. Kalker, intègre le signal dans l'espace caractéristique d'une transformée (e.g. DFT, DCT, Ondelette). Elle fournit des coefficients perceptivement significatifs, fiables pour le tatouage tel que conceptualisé dans [4, Sec. 8.1.3].

Le tatouage connaît un regain d'intérêt grâce aux progrès de l'apprentissage profond. De nouvelles méthodes améliorent la robustesse à un large éventail d'altérations grâce à des réseaux neuronaux offrant un espace latent fiable où intégrer l'information. Des exemples sont le marquage direct dans l'espace sémantique résultant d'un apprentissage supervisé sur un ensemble donné de classes comme ImageNet [15], ou l'apprentissage explicite d'un réseau de tatouage pour qu'il soit invariant à un ensemble de perturbations de l'image. Dans ce cas, les réseaux sont généralement des architectures encodeur-décodeur entraînées de bout en bout pour le tatouage [19, 17, 2, 18].

Notre idée maîtresse est de tirer parti des propriétés des réseaux *auto-supervisés* pour tatouer les images. Idéalement, selon [4], un coefficient perceptuellement significatif ne change pas à moins que le contenu visuel de l'image soit différent. Certaines méthodes auto supervisées visent précisément à créer des représentations invariantes aux augmentations, sans connaissance explicite de la sémantique de l'image [3, 8]. Ces réseaux pré entraînés offrent "gratuitement" l'espace latent désiré, ce qui évite un entraînement lourd comme HiDDeN [19].

Afin de marquer de manière robuste dans les espaces latents, une descente de gradient est effectuée sur l'image. Pour garantir la robustesse et l'imperceptibilité des tatouages, nous incluons l'augmentation des données et le prétraitement des images au moment du marquage.

Nos contributions sont les suivantes :

— Nous fournissons un algorithme de tatouage numérique qui peut encoder à la fois des marques et des messages binaires dans les espaces latents de tout réseau pré-entraîné ;

— Nous montrons comment tirer parti de filtres perceptuels et de l'augmentation des données au moment du marquage ;

— Nous montrons expérimentalement que les réseaux entraînés avec auto-supervision fournissent d'excellents espaces de marquage.

2 Technique de tatouage

2.1 Réseaux auto-supervisés

Motivation. Nous désignons l'espace image par $\mathcal{I}$ et l'espace latent d'un réseau de neurones par $\mathcal{F} = \mathbb{R}^d$.

Notre hypothèse est que le SSL (*Self-Supervised Learning*) produit d'excellents espaces d'intégration car (1) il entraîne explicitement les caractéristiques pour qu'elles soient invariantes à l'augmentation des données ; et (2) il ne souffre pas de l'*effondrement sémantique* qu'on peut observer en classification supervisée et qui se débarrasse de toute information qui n'est pas nécessaire pour assigner des classes [6]. Parmi les méthodes de SSL de la littérature, nous choisissons DINO [3] pour sa vitesse d'apprentissage et ses performances en recherche par similarité.

Pré-entraînement avec DINO. L'auto distillation sans étiquettes DINO [3] (*self-distilation with no labels*) entraîne un réseau élève à correspondre aux sorties d'un réseau enseignant sur des vues différentes de la même image. L'invariance des caractéristiques est assurée par des augmentations aléatoires pendant l'apprentissage : transformations valuemétriques (modification de couleur, flou gaussien, solarisation) et géométriques (recadrages).

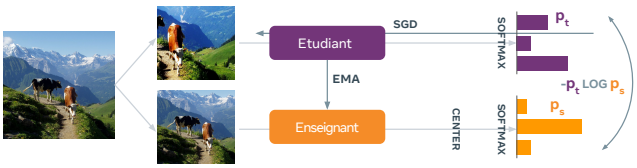

FIGURE 1 – Entrainement auto-supervisé avec DINO [3]

Blanchiment par ACP. L'intégration du tatouage pousse les caractéristiques de l'image vers une région spatiale arbitraire (définie par une clé secrète et le message à cacher). Il est essentiel qu'elles ne soient pas concentrées sur une variété éloignée de cette région. Pour pallier à ce problème, les caractéristiques de sortie sont transformées par blanchiment ACP (ou PCA-whitening). Cette transformation linéaire produit des vecteurs de dimension $d = 2048$, centrés, avec une covariance unitaire.

2.2 Tatouage par rétropropagation

Le marquage prend une image originale I_o et produit une image visuellement similaire I_w. Dans l'espace image $\mathcal{I}$, la distorsion est mesurée par l'erreur quadratique : $\mathcal{L}_i(I_w, I_o) = \|I_w - I_o\|^2$ ou de façon équivalente par le PSNR (Peak Signal to Noise Ratio).

Dans l'espace des caractéristiques $\mathcal{F}$, nous définissons une région $\mathcal{D}$ qui dépend d'une clé secrète (configurations zéro-bit et multi-bit) et du message à cacher (seulement dans la configuration multi-bit). Sa définition est reportée à la Sec. 2.3 ainsi que l'objectif $\mathcal{L}_w : \mathcal{F} \to \mathbb{R}$ qui capture la distance d'une caractéristique $x \in \mathcal{F}$ par rapport à $\mathcal{D}$. Nous définissons également un ensemble $\mathcal{T}$ d'augmentations, qui incluent rotation, recadrage, flou, etc., chacune avec une plage de paramètres. $\mathrm{Tr}(I, t) \in \mathcal{I}$ désigne l'application de la transformation $t \in \mathcal{T}$ à I.

Les pertes $\mathcal{L}_w$ et $\mathcal{L}_i$ sont combinées en :

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

Le terme $\mathcal{L}_w$ vise à pousser la caractéristique de *toute transformation* de I_w dans $\mathcal{D}$, tandis que le terme $\mathcal{L}_i$ favorise une faible distorsion.

La méthode d'optimisation est typique de la littérature sur les attaques adversariales [13] :

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

où $\mathcal{C}(I_o) \subset \mathcal{I}$ est l'ensemble des images admissibles en vue des contraintes perceptuelles : SSIM [16]≥ 0 en tout pixel et PSNR$\geq$cible). La minimisation de l'objectif est effectuée par descente de gradient stochastique.

2.3 Détection et décodage

Nous envisageons deux scénarios : le tatouage zéro-bit (détection uniquement) et le tatouage multi-bits (décodage du message caché).

Zero-bit. À partir d'une clé secrète $a \in \mathcal{F}$ s.t. $\|a\| = 1$, la région de détection pour $\phi(I)$ est l'hypercône dual :

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

Le taux de faux positifs (FPR) est donné par :

$$\begin{aligned} \mathrm{FPR} &:= \mathbb{P}\left(\phi(I) \in \mathcal{D} \mid \text{"clef } a \text{ uniformément distribuée"}\right) \\ &= 1 - I_{\cos^2(\theta)}\left(\frac{1}{2}, \frac{d-1}{2}\right) \end{aligned} \quad (4)$$

où $I_\tau(\alpha, \beta)$ est la fonction Beta incomplète régularisée. Le tatouage est obtenu en maximisant l'objectif :

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

Cette quantité est négative lorsque $x \notin \mathcal{D}$ et positive sinon. À l'origine, I. Cox l'appelait l'estimation de la robustesse [4, Sec. 5.1.3]. Ce détecteur est optimal sous la configuration gaussienne asymptotique [7].

Multi-bit. Nous supposons maintenant que le message à cacher est $m = (m_1, ..., m_k) \in \{-1, 1\}^k$. Ici, la clé secrète est une famille orthogonale $a_1, ..., a_k \in \mathbb{R}^d$ de porteuses échantillonnées aléatoirement. Nous modulons m dans les signes de la projection de la caractéristique $\phi(I)$ contre chacune des porteuses a_k, de sorte que le décodeur s'écrit :

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

Au moment du marquage, la fonctionnelle est maintenant définie comme une marge maximale avec $\mu \geq 0$ sur les 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 Expériences & résultats

3.1 Configuration expérimentale

Data. Nous évaluons notre méthode sur : 1000 images du jeu de données YFCC100M [14] pour la variété de son contenu, CLIC [1] composé de 118 images haute résolution pour la comparaison avec [15], et 1000 images de MS-COCO [11] pour la comparaison avec [19, 12].

Pré-entraînement. Nous utilisons la dernière couche ($d = 2048$) de l'architecture ResNet-50 [9] pour extraire les caractéristiques des images. Le réseau est entraîné sur ILS-VRC2012 [5], en utilisant 200 époques d'apprentissage auto supervisé DINO. Le blanchiment ACP est appris sur 100k images distinctes de YFCC (resp. COCO) lors de l'évaluation sur YFCC et CLIC (resp. COCO).

Intégration du tatouage. Nous fixons d'abord un PSNR cible et, dans le cas zéro-bit, un FPR qui définit l'angle d'hypercône θ. L'optimisation (2) utilise Adam [10] sur 100 itérations avec un taux d'apprentissage de $0,01$. Le poids dans (1) est fixé à $\lambda = 1$ (zéro bit) ou $\lambda = 5 \cdot 10^4$ (plusieurs bits). La marge de (7) est fixée à $\mu = 5$.

À chaque itération, des contraintes sont imposées à I (filtres SSIM et PNSR). Ensuite, une transformation t est choisie aléatoirement dans $\mathcal{T}$ (identité, rotation, flou, recadrage ou redimensionnement). Puis les paramètres de la transformation sont tirés aléatoirement.

3.2 Tatouage zéro-bit

Influences de l'auto-supervision et de l'augmentation. La mesure de performance est le taux de vrais positifs (TPR), pour un PSNR=40dB, et un FPR=10^{-6}. La Fig. 2 évalue la robustesse pour le cas spécifique de la rotation. La rotation est nécessaire à la fois aux étapes de pré entraînement et de marquage pour obtenir une robustesse élevée contre celle-ci. La comparaison sur une plus large gamme de transformations est donnée dans le tab. 1.

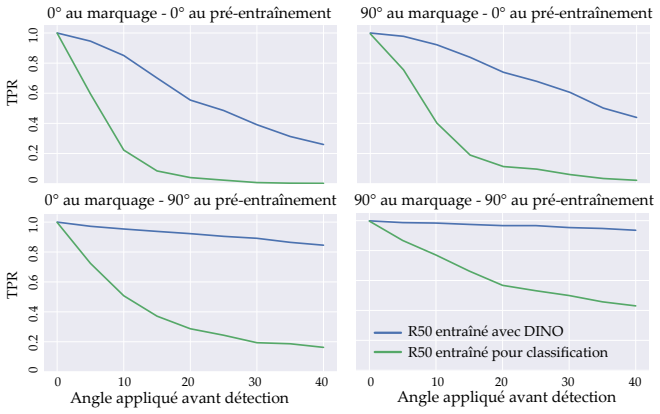

FIGURE 2 – Robustesse face à la rotation. Chaque ligne (colonne) représente différentes amplitudes de rotation à l'entraînement (resp. au marquage).

TABLE 1 – TPR sur diverses attaques. 1ère configuration : performance avec SSL par rapport aux réseaux supervisés. 2nde configuration : ($\star$) meilleurs résultats dans [15], ($\star\star$) notre implémentation de [15]. † indique les augmentations utilisées lors du pré-entraînement.

	Config. 1 : YFCC		Config. 2 : CLIC		
Transformations	SSL	Sup.	Nôtre	[15] ($\star$)	[15] ($\star\star$)
Identité	1.00†	1.00†	1.00†	1.0†	1.00†
Rotation (25)	0.97†	0.54†	**1.00**†	≈ 0.3†	0.27†
Recadrage (0.5)	0.95†	0.79†	1.00†	≈ 0.1†	1.00†
Recadrage (0.1)	0.39†	0.06†	**0.98**†	≈ 0.0†	0.02†
Redimens. (0.7)	0.99†	0.85†	1.00†	-	1.00†
Flou (2.0)	0.99†	0.04	**1.00**†	-	0.25
JPEG (50)	0.81	0.20	0.97	≈ 1.0	0.96
Luminosité (2.0)	0.94†	0.71	0.96†	-	0.99
Contraste (2.0)	0.96†	0.65	1.00†	-	1.00
Teinte (0.25)	1.00†	0.46	1.00†	-	1.00
Meme	0.99	0.94	1.00	-	0.98
Screenshot	0.76	0.18	**0.97**	-	0.86

Résultats qualitatifs. La Fig. 3 présente une image tatouée à PSNR 40dB et certaines altérations détectées, ainsi que l'amplitude du signal de tatouage. Le tatouage est presque invisible, même pour un œil exercé, car il est ajouté dans les régions texturées en raison du filtre perceptif SSIM appliqué pendant le tatouage.

Comparaison avec l'état de l'art. Le tab. 1 compare notre méthode avec [15] sur des images haute résolution issues de CLIC. Le FPR est fixé à 10^{-3} et le PSNR doit être $\geq$ 42dB. Nous observons une forte amélioration par rapport à [15], notamment pour les grandes rotations, les recadrages et le flou gaussien où notre méthode donne une détection presque parfaite sur les 118 images.

3.3 Tatouage multi-bit

Résultats quantitatifs. Nous évaluons notre méthode sur YFCC, avec un PSNR=40dB et une charge k de 30 bits aléatoires comme dans [19, 12]. Le tab. 2 présente le taux d'erreur sur les bits (BER) et les mots (WER) pour diffé-

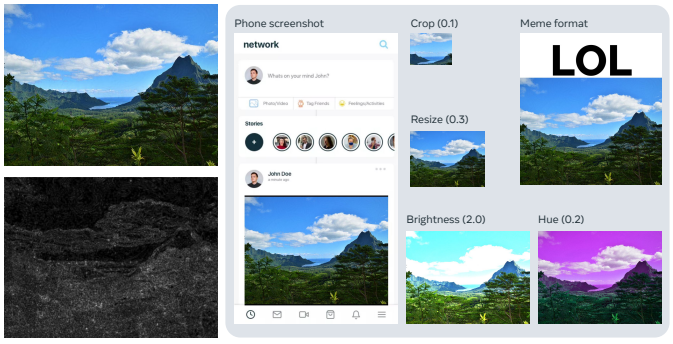

FIGURE 3 – Exemple d'une image (800×600) tatouée à PSNR=40dB et FPR=10^{-6}, et quelques altérations pour lesquelles le tatouage est détecté. L'image en noir et blanc montre l'amplitude du signal du tatouage.

TABLE 2 – BER et WER (%) pour un codage de 30-bits à PSNR=40dB.

Transform.	Id.	Rot. (25)	Recad. (0.5)	Recad. (0.1)	Redim. (0.7)	Flou (2.0)	JPEG (50)	Lumi. (2.0)	Contr. (2.0)	Teinte (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 – Comparaison des BER. La 1ere ligne utilise les résolutions originales de COCO, tandis que les autres utilisent une version redimensionnée (à 128×128). Les résultats pour [19, 12] proviennent de [12]. ‡ indique les transformations utilisées dans le processus de tatouage.

Transformation	Identity	JPEG (50)	Flou (1.0)	Recad. (0.1)	Redi. (0.7)	Teinte (0.2)
Nôtre	$0.00^{\ddagger}$	0.04	$0.00^{\ddagger}$	$0.18^{\ddagger}$	$0.00^{\ddagger}$	0.03
Nôtre, 128×128	$0.00^{\ddagger}$	0.16	$0.01^{\ddagger}$	$0.45^{\ddagger}$	$0.18^{\ddagger}$	0.06
HiDDeN [19]	$0.00^{\ddagger}$	$0.23^{\ddagger}$	$0.01^{\ddagger}$	$0.00^{\ddagger}$	0.15	0.29
Dist. Agnostic [12]	$0.00^{\ddagger}$	$0.18^{\ddagger}$	$0.07^{\ddagger}$	$0.02^{\ddagger}$	0.12	0.06

rentes attaques. Le décodage atteint de faibles taux sur une large gamme d'attaques géométriques (rotation, recadrages, redimensionnement, etc.) et valuemétriques (luminosité, teinte, contraste, etc.). La rotation et le flou gaussien sont particulièrement inoffensifs puisqu'ils sont vus à la fois au moment du pré-entraînement et du marquage.

Comparaison avec l'état de l'art. En 3 nous comparons notre méthode à deux méthodes encodeur-décodeur [19, 12], en utilisant leurs paramètres : une charge de 30 bits, un PSNR cible de 33dB, sur 1000 images de COCO redimensionnées à 128×128. Dans l'ensemble, notre méthode donne des résultats comparables à l'exception du recadrage. Une explication est que le processus itératif (autour de 100 itérations) ne couvre pas suffisamment de recadrages possibles de l'image. En revanche, notre méthode atteint des performances similaires pour la compression JPEG sans entraînement spécifique (alors [19, 12] sont entraînées pour). De plus, notre méthode s'adapte facilement aux images de plus haute résolution (cf. tab. 2), alors que les méthodes [19, 12] nécessiteraient un entraînement spécifique pour une résolution donnée.

4 Conclusion

Cet article propose un moyen d'intégrer de manière robuste et invisible des tatouages dans des images numériques, en tatouant dans les espaces latents de réseaux auto-supervisés "prêts à l'emploi". Notre méthode de tatouage à zéro bit améliore grandement les performances par rapport à la méthode mère [15]. Lorsque nous étendons la méthode au tatouage multi-bit, nous obtenons des résultats prometteurs, comparables à l'état de l'art, et même meilleurs en ce qui concerne certaines transformations de l'image (compression JPEG ou flou).

Plus intéressant encore, les réseaux entraînés par auto-supervision génèrent naturellement d'excellents espaces de tatouage, sans être explicitement entraînés à le faire. Cependant, par rapport aux techniques de tatouage profond de type encodeur-décodeur, le tatouage d'images avec notre méthode est coûteux puisque le processus est itératif, et nécessite un GPU. Dans des travaux futurs, nous espérons montrer qu'une adaptation plus poussée du réseau pour la tâche spécifique du tatouage améliorerait les performances et l'efficacité.

Références

[1] Workshop and challenge on learned image compression.

[2] 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.

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

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

[5] 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.

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

[7] 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.

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

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

[11] 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.

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

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

[14] 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.

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

[16] 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.

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

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

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