SUPPLEMENTARY MATERIAL TO: MONSTERS IN THE DARK: SANITIZING HIDDEN THREATS WITH DIFFUSION MODELS

Anonymous authors
Paper under double-blind review

In this supplementary material, dive deeper into the steganographic techniques used in this work and present additional sanitization results.

1 STEGANOGRAPHY DETAILS

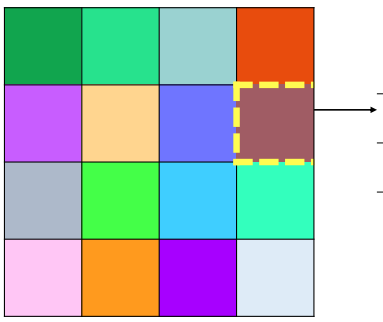
1.1 LEAST SIGNIFICANT BIT METHOD

Digital images are a commonly used cover medium, which are composed of pixels, or a finite set of digital values in a two-dimensional array. While there are various models for representing colors in pixels, the RGB color model utilizes red, green, and blue color channels. Each pixel of an image is then composed of three 8-bit values which represent the intensity of each color, as shown in Figure 1. Here, the RGB value of the indicated pixel is RGB(160, 92, 100) or represented in binary RGB(10100000, 01011100, 01100100). In binary, the leftmost bit of the binary digit is known as the most significant bit, and the rightmost bit is known as the least significant bit. This results from the degree of change that can occur in the 8-bit binary digit value if this bit is switched. For example, if the leftmost bit of 11111111 (or 255) is flipped, the number becomes 0111111 (or 127), changing by approximately 50%. If, however, the rightmost bit is changed from 11111111 (255) to 11111110 (254), the value only changes by approximately 0.4%. In this way, information can be embedded in the least significant bits of an RGB value while imperceptibly changing the visual rendering of the image. Figure 2 shows the effects of flipping the two least significant bits of every color channel in the pixels of the original image. The resulting difference between the two images are imperceptible to the Human Visual System (HVS).

For images of the same size, it is common to hide the four most significant bits of the secret image in the four least significant bits of the cover image, which was first introduced in Kurak & McHugh (1992). As such, this work makes use of the four least significant bits to hide secrets via the LSB implementation.

1.2 DEPENDENT DEEP HIDING

Dependent deep hiding (DDH) differs from a traditional hiding method in that it utilizes a deep learning model for the hide and reveal functions, as shown in figure 3a. The hide network takes as



Channel	Decimal	Binary
R	160	10100000
G	92	01011100
B	100	01100100

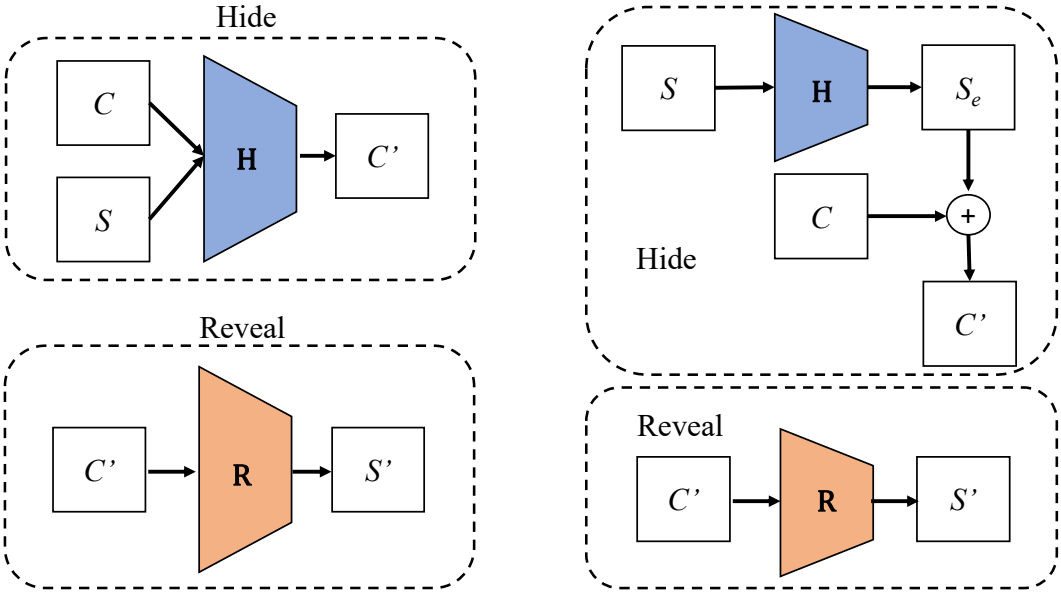
Figure 1: Image autopsy demonstrating the red, green, and blue color channels of a single pixel in an image.

(a) Original Image (b) Altered Image

Figure 2: A demonstration of the LSB method for hiding images. The original image 2a appears identical to be the altered image 2b. The altered image shows the effects of flipping the two least significant bits of each channel (red, green, and blue) of every pixel in the image.

input a cover and a secret and produces a container image. The reveal network then takes as input the produced container and maps this image to a revealed secret. The hide and reveal models are usually trained in tandem, and the loss function minimizes the difference between the container and cover, as well as the revealed secret and original secret. Recent CNN-based and GAN-based methods include SGAN Volkhonskiy et al. (2020), UT-GAN Yang et al. (2019), ASDL-GAN Tang et al. (2017), SPAR-RL Tang et al. (2020), R-GAN Wu et al. (2020), HIDDeN Zhu et al. (2018), SSteGAN Wang et al. (2018), and Deep Steganography Baluja (2017). This work utilizes a CNN-based implementation available from a code base that also contains a universal deep hiding implementation[1]. This code base was chosen because of its dual (UDH and DDH) functionality. By utilizing implementations from the same code base, we are better able to accurately compare the methodologies (DDH vs. UDH) as differences in programming implementations can affect training performance Henderson et al. (2018).

[1] https://github.com/ChaoningZhang/Universal-Deep-Hiding



(a) Deep Dependent Hiding

(b) Universal Deep Hiding

Figure 3: A comparison between dependent deep hiding (DDH) and universal deep hiding (UDH). In 3a, a cover C and secret S are used as inputs to the hide network $\mathcal{H}$, producing a container image C'. The reveal network of DDH takes a container as an input and maps this image to the revealed secret S'. While DDH is cover dependent, UDH only relies on the secret. The secret, therefore, acts as the only input into the hide network $\mathcal{H}$ in 3b, which produces a secret noise image S_e. S_e can then be added to any arbitrary cover image to produce a container. The reveal network $\mathcal{R}$ of UDH maps this container to a revealed secret, which resembles the original secret.

1.3 Universal Deep Hiding

Universal deep hiding (UDH) is similar to DDH in that it also utilizes deep learning, but UDH is cover independent Zhang et al. (2020). In figure 3b, the input to the hide network is just the secret, which produces a corresponding secret noise image S_e. S_e is then added with an arbitrary cover to produce a container image. A reveal network is then used to reproduce the original secret. The secret noise image can be applied to any cover, and the secret is still retrievable by the reveal network. The hide and reveal networks are trained in tandem in UDH as well.

2 Image Metrics

The equations for each of these metrics (MSE, PSNR, SSIM) are shown in equations 1, 2, and 3, where A and B are the compared images of size (c, h, w), MAX is the maximum possible pixel value (for a given bit depth), μ_A and μ_B are the average values of images A and B, σ_A^2 and σ_B^2 are the variances of images A and B, σ_{AB} is the covariance of A and B, and c_1 and c_2 are constants to avoid division by zero.

$$MSE(A,B) = \frac{1}{chw}\sum_{i=1}^{c}\sum_{j=1}^{h}\sum_{k=1}^{w}(A_{i,j,k} - B_{i,j,k})^2 \tag{1}$$

$$PSNR(A,B) = 10\log_{10}\left(\frac{MAX^2}{MSE(A,B)}\right) \tag{2}$$

$$SSIM(A,B) = \frac{(2\mu_A\mu_B + c_1)(2\sigma_{AB} + c_2)}{(\mu_A^2 + \mu_B^2 + c_1)(\sigma_A^2 + \sigma_B^2 + c_2)} \tag{3}$$

3 SANITIZATION

The following section includes larger images of the images included in the main body of the paper (fig. 2) or more images of the same analysis.

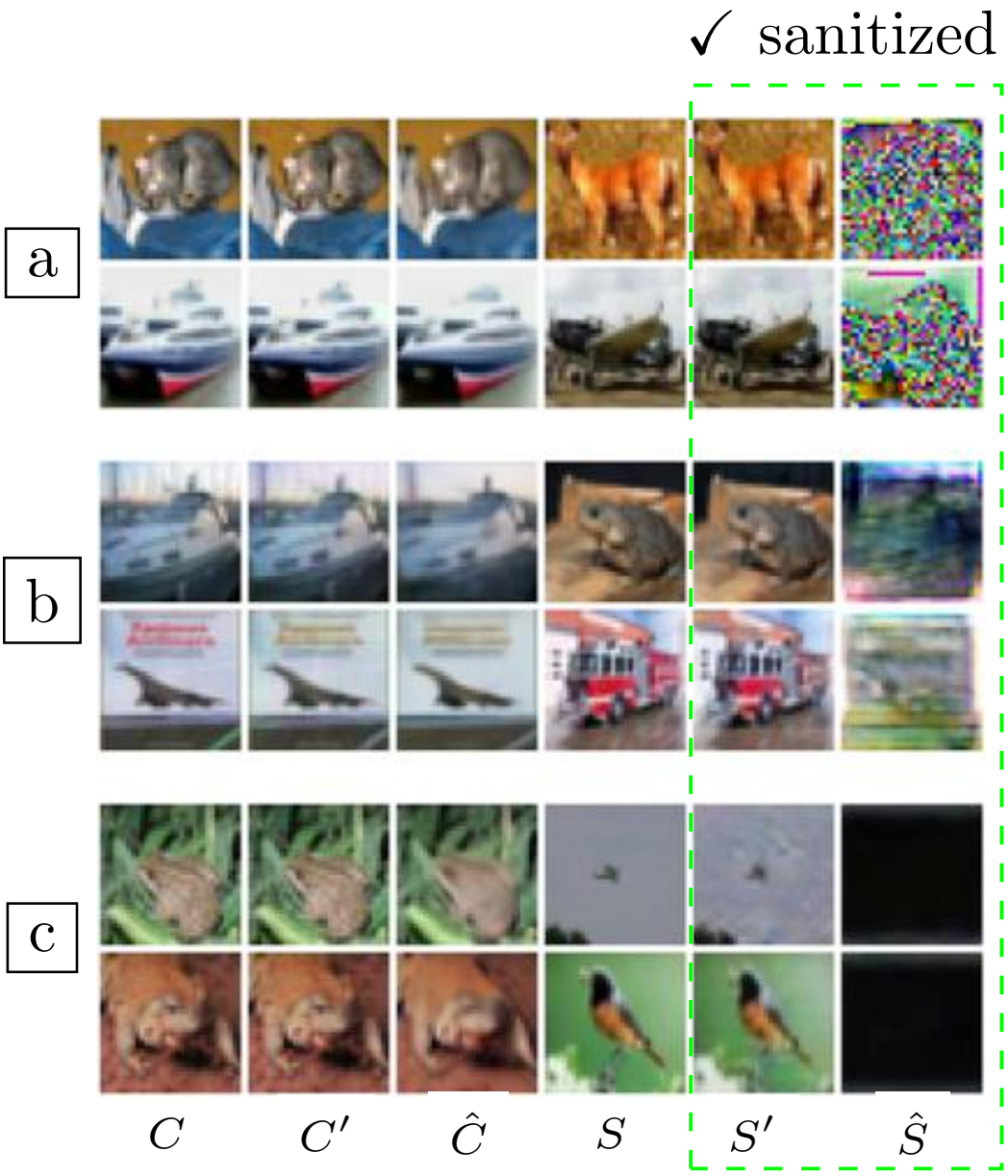


Figure 4: DM-SUDS sanitization where a) are containers hidden with LSB, b) DDH, and c) UDH.

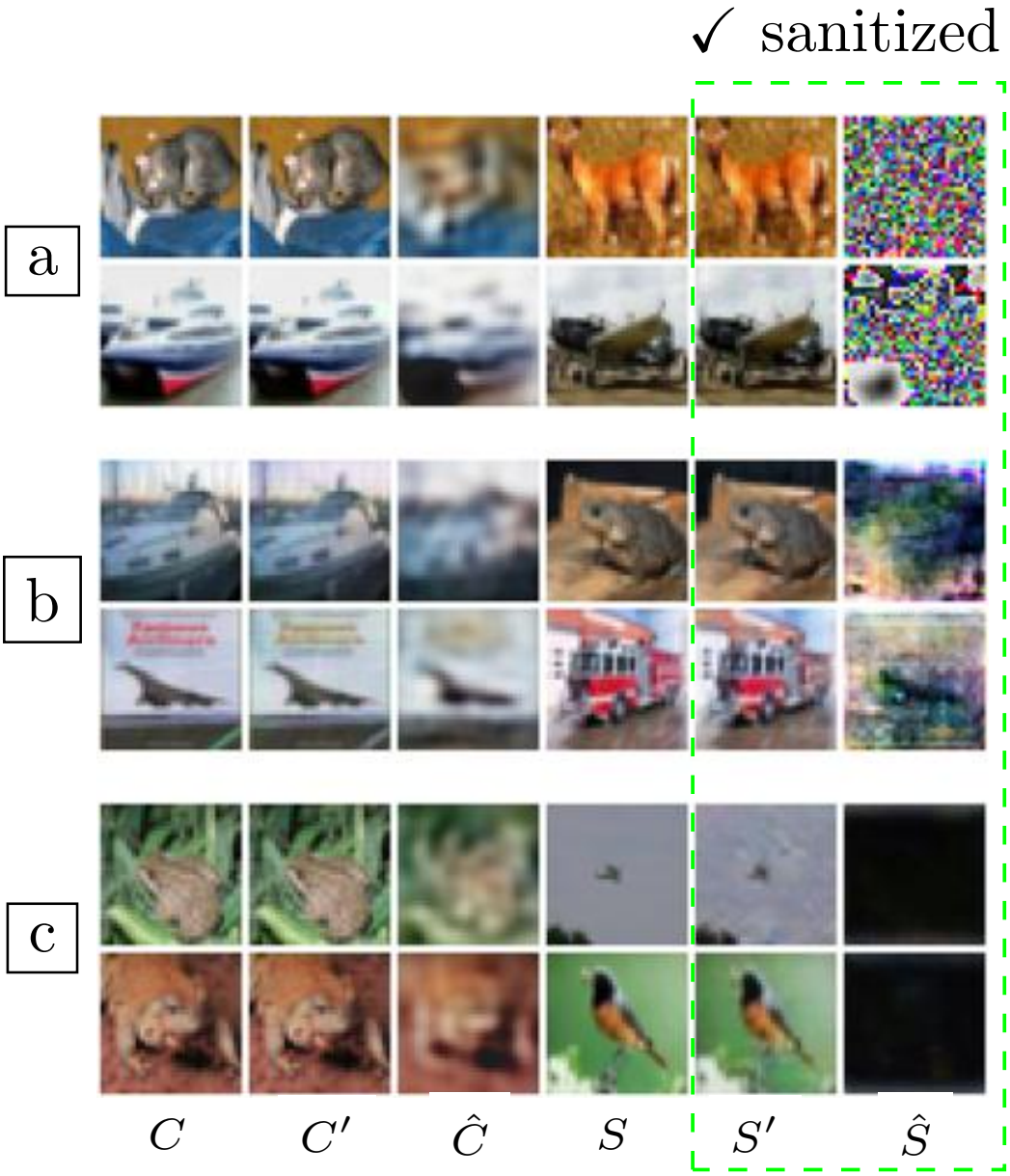


Figure 5: SUDS sanitization where a) are containers hidden with LSB, b) DDH, and c) UDH.

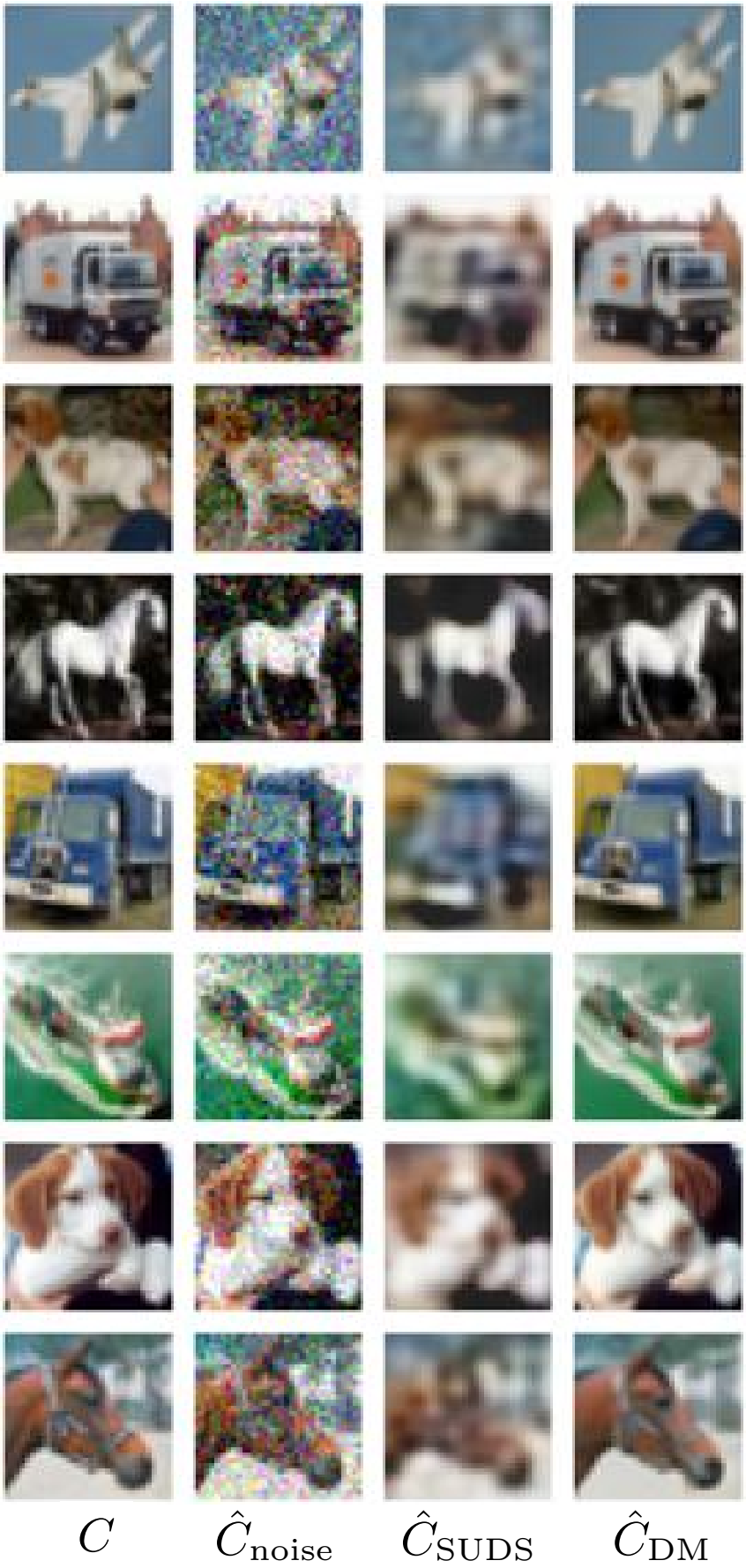


Figure 6: Comparison among the image preservation capabilities of Gaussian noise, SUDS, and DM-SUDS sanitization techniques.

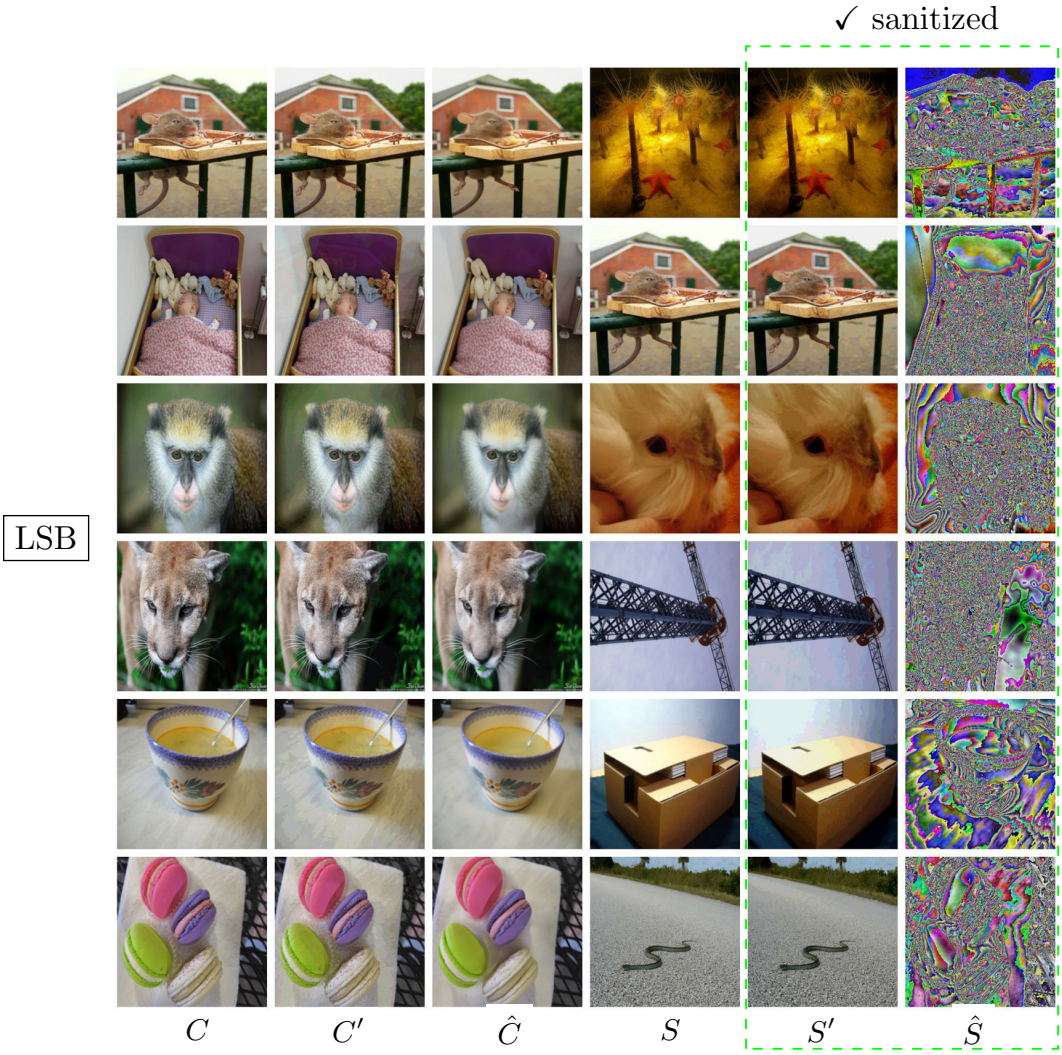


Figure 7: DM-SUDS capabilities on the ImageNet dataset.

REFERENCES

Shumeet Baluja. Hiding images in plain sight: Deep steganography. *Advances in neural information processing systems*, 30, 2017. https://papers.nips.cc/paper/2017/file/838e8afb1ca34354ac209f53d90c3a43-Paper.pdf.

Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018. https://ojs.aaai.org/index.php/AAAI/article/view/11694/11553.

C. Kurak and J. McHugh. A cautionary note on image downgrading. In *[1992] Proceedings Eighth Annual Computer Security Application Conference*, pp. 153–159, 1992. doi: 10.1109/CSAC.1992.228224. https://ieeexplore.ieee.org/document/228224.

Weixuan Tang, Shunquan Tan, Bin Li, and Jiwu Huang. Automatic steganographic distortion learning using a generative adversarial network. *IEEE Signal Processing Letters*, 24(10):1547–1551, 2017. https://ieeexplore.ieee.org/document/8017430.

Weixuan Tang, Bin Li, Mauro Barni, Jin Li, and Jiwu Huang. An automatic cost learning framework for image steganography using deep reinforcement learning. *IEEE Transactions on In-*

formation Forensics and Security, 16:952–967, 2020. https://ieeexplore.ieee.org/document/9205850.

Denis Volkhonskiy, Ivan Nazarov, and Evgeny Burnaev. Steganographic generative adversarial networks. In *Twelfth international conference on machine vision (ICMV 2019)*, volume 11433, pp. 991–1005. SPIE, 2020. https://doi.org/10.1117/12.2559429.

Zihan Wang, Neng Gao, Xin Wang, Xuexin Qu, and Linghui Li. Sstegan: Self-learning steganography based on generative adversarial networks. In *International Conference on Neural Information Processing*, pp. 253–264. Springer, 2018. https://link.springer.com/chapter/10.1007/978-3-030-04179-3_22.

Haibin Wu, Fengyong Li, Xinpeng Zhang, and Kui Wu. Gan-based steganography with the concatenation of multiple feature maps. In *International Workshop on Digital Watermarking*, pp. 3–17. Springer, 2020. https://link.springer.com/chapter/10.1007/978-3-030-43575-2_1.

Jianhua Yang, Danyang Ruan, Jiwu Huang, Xiangui Kang, and Yun-Qing Shi. An embedding cost learning framework using gan. *IEEE Transactions on Information Forensics and Security*, 15: 839–851, 2019. https://ieeexplore.ieee.org/abstract/document/8735922.

Chaoning Zhang, Philipp Benz, Adil Karjauv, Geng Sun, and In So Kweon. Udh: Universal deep hiding for steganography, watermarking, and light field messaging. *Advances in Neural Information Processing Systems*, 33:10223–10234, 2020. https://proceedings.neurips.cc/paper/2020/file/73d02e4344f71a0b0d51a925246990e7-Paper.pdf.

Jiren Zhu, Russell Kaplan, Justin Johnson, and Li Fei-Fei. Hidden: Hiding data with deep networks. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 657–672, 2018. https://link.springer.com/chapter/10.1007/978-3-030-01267-0_40.