Supplementary Material: Zero-Shot Detection of AI-Generated Images

Davide Cozzolino[1], Giovanni Poggi[1], Matthias Nießner[2], and Luisa Verdoliva[1,2]

[1] University Federico II of Naples, 80125 Naples, Italy
[2] Technical University of Munich, 85748 Garching, Germany
{davide.cozzolino,poggi,verdoliv}@unina.it, niessner@tum.de

In this supplementary document, we give more details about the comparison methods and the evaluation datasets present in the main paper. Moreover, we report additional ablation results and experimental analysis.

1 Reference methods

In our comparative analysis, in order to ensure a fair comparison, we include only reference techniques with code and/or pre-trained models publicly available online. Eventually, we considered the following methods, listed approximately from less recent to most recent:

1. **Wang2020** [12] it is based on a plain ResNet50 backbone and represents a reference in the research community. Its main peculiarity is the training phase, based on a large dataset of 360k real (LSUN) and 360k synthetic (ProGAN) images, the latter taken from 20 different classes. Augmentation based on compression and blurring is also used. The proposed dataset has been widely adopted in subsequent papers to train new models.
2. **PatchFor** [2] is a fully convolutional classifier on an XceptionNet backbone, with a limited receptive field that allow to focus on image local patches rather than on the global structure. The patch-based predictions are also used to visualize patterns that indicate the regions where real and fake image can be easily detected. Finally, images are properly pre-processed to avoid learning image formatting artifacts.
3. **Liu2022** [6] tries to distinguish real images from synthetic images based on statistical differences observed in their learned noise patterns. To improve performance, spatial information is then combined with information gathered in the frequency domain.
4. **Corvi2023** [3] it relies on a modified ResNet50 architecture where subsampling steps are mostly avoided to preserve forensic traces. Also resizing is not carried out both in the training and testing phases to preserve subtle forensic traces. It is trained on a dataset of latent diffusion images, with strong augmentation to gain higher robustness and generalization ability.

5. **LGrad** [11] relies exclusively on low-level traces and thus removes the image content altogether, by extracting only the image gradients through a pre-trained CNN model. In the noise residual domain, real and synthetic images are distinguished based on their different inter-pixel dependencies.

6. **DIRE** [13] is an inversion-based method, relying on the assumption that synthetic images can be well approximated by a generator while real images cannot. To perform detection, a ResNet50 backbone is fed with the difference between the image and its version obtained by the inversion process.

7. **DE-FAKE** [9] leverages an image captioning model (BLIP) to generate the textual prompt of the image. The high-level embeddings of image and text, extracted using a large pre-trained model (CLIP), are then concatenated and fed to a multilayer perceptron for the binary classification.

8. **Ojha2023** [7] uses features extracted by a large pre-trained model, CLIP, to detect synthetic images. After feature extraction, the dataset proposed in [12] is used to design the classifier, testing various strategies, from nearest neighbor to linear probing.

9. **NPR** [10] builds on the widespread use of up-sampling operations in generation models. proposing a simple way to represent up-sampling artifacts called Neighboring Pixel Relationships (NPR). NPR is computed from the difference between the original image and its interpolated version, which is then fed into a ResNet-50 trained on only four categories of the dataset proposed in [12].

10. **AEROBLADE** [8] is a training-free approach that is based on the intuition that synthetic images are more accurately reconstructed by the autoencoder than real images. Then the discrimination is carried out by evaluating the reconstruction error of the autoencoder measured using the LPIPS distance between the image and its reconstructed version.

2 Datasets

Table 1 lists all generators of synthetic images used in our experiments. For each generator we provide the corresponding real data used in the test set, the size of the generated images and the number of images used in the experiments. Overall, we are considering a large variety of synthetic generators of widely different characteristics for a total of 29k synthetic images and 6k real images. These are a collection of datasets proposed in [1, 3, 5, 7, 12] and generated by ourselves. It is worth noting that synthetic and real images are characterized by the same semantic content to avoid polarization. We also considered different reals from the same generator to understand the influence of the pristine class.

Table 1: Details of datasets used in all experiments: synthetic image generator, real images used in testing, number and size of images, source.

generator	real data	# images	image size	source
GauGAN	COCO	1000	256^2	[12]
BigGAN	ImageNet	1000	256^2,512^2	[3]
StarGAN	FFHQ	1000	256^2	ours
StyleGAN2	LSUN	500	256^2	[3]
	FFHQ	500	256^2,1024^2	[3]
GigaGAN	ImageNet	500	256^2	[5]
	COCO	500	512^2	[5]
Diff.GAN	LSUN	1000	256^2	ours
GALIP	COCO	1000	256^2	ours
DALL·E	LAION	1000	256^2	[7]
DDPM	FFHQ	1000	256^2	ours
ADM	LSUN	500	256^2	[3]
	ImageNet	500	256^2	[3]
GLIDE	COCO	1000	256^2	[3]
	RAISE	1000	256^2	[1]
	LAION	3000	256^2	[7]
DiT	ImageNet	1000	256^2,512^2	ours
Stable D. 1.4	COCO	1000	256^2	ours
	RAISE	1000	512^2	[1]
Stable D. 2	COCO	1000	256^2-768^2	ours
	RAISE	1000	973^2-1024^2	[1]
SDXL	COCO	1000	1024^2	ours
	RAISE	1000	973^2-1024^2	[1]
Deep.-IF	COCO	1000	1024^2	ours
DALL·E 2	COCO	1000	1024^2	[3]
	RAISE	1000	1024^2	[1]
DALL·E 3	COCO	1000	1024^2	ours
	RAISE	1000	1024^2-1355^2	[1]
Midjourney	RAISE	1000	1024^2-1104^2	[1]
Adobe Firefly	RAISE	1000	2032^2-2048^2	[1]

3 Additional ablation study

Here we describe a set of experiments designed to gain some insight into a few questions concerning training and testing:

Q1: how much do training and testing datasets impact on performance?
Q2: how important is their alignment?
Q3: which are the best decision statistics?

To this end, we trained the lossless coder on three different datasets of real images: Open Images, LAION, and COCO. In all cases, we considered two versions, with and without data augmentation. Then we used six different testing datasets. Each one includes the real images taken, in turn, from ImageNet, LAION, LSUN,

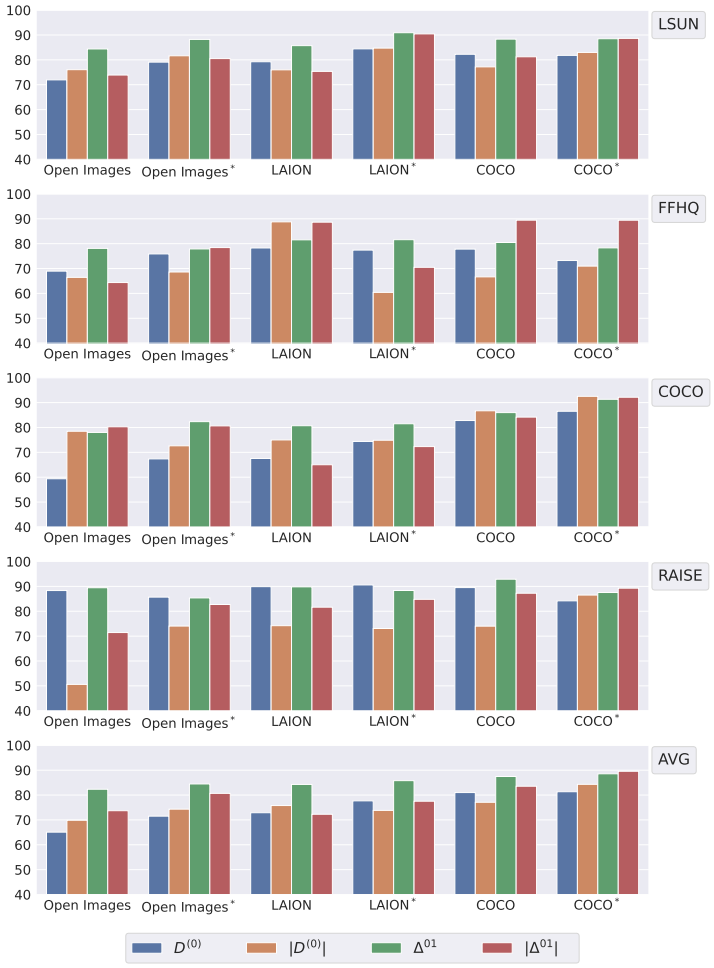

Fig. 1: Expanded ablation study, completing the results of figure 6 of main paper. For each testing dataset (chart), and each training dataset (group of bars), we compute the AUC of the proposed method with four decision statistics (individual bars). In each testing dataset, synthetic images are selected to match the corresponding real images. The final chart shows the average over all testing datasets.

FFHQ, COCO, and RAISE, with the associated fakes as outlined in Tab.1. For example, the FFHQ testing dataset includes StarGAN, StyleGAN2, and DDPM synthetic images. In all cases, we tested the four decision statistics defined in the main paper.

Results are reported in six charts, one for each testing dataset, two of them already shown in Fig.6 of the main paper and four more in Fig.1, followed by a final chart with average results. From these results we see that, as expected, the quality of the training set has a significant impacts on performance. By

Table 2: Results in terms of AUC averaged for each real dataset. We have both JPEG compressed real images (COCO, IMAGENET, LSUN, LAION) and uncompressed images (FFHQ, RAISE).

Real data	Wang2020	PatchFor.	Lin2022	Corvi2023	LGrad	DIRE	DE-FAKE	Ojha2023	NPR	AEROBLADE	Ours $D^{(0)}$	Ours $\lvert D^{(0)}\rvert$	Ours Δ^{01}	Ours $\lvert\Delta^{01}\rvert$
COCO	77.3	74.8	93.9	94.8	73.9	**99.7**	85.8	92.2	91.5	82.7	86.5	92.5	91.3	92.2
IMAGENET	71.5	77.9	93.6	78.8	70.6	**99.7**	69.9	89.3	81.5	69.9	85.5	82.9	89.6	85.9
LSUN	85.2	81.3	89.9	88.6	93.5	53.4	40.4	94.1	**98.0**	42.3	81.8	83.0	88.5	89.0
LAION	69.5	72.9	92.9	92.6	92.8	**99.9**	58.1	96.3	**99.7**	46.4	76.8	90.0	96.0	91.9
FFHQ	73.7	**94.5**	74.2	84.8	55.5	39.7	48.0	89.5	81.8	75.3	73.2	71.0	78.3	89.4
RAISE	46.9	58.2	39.9	**89.3**	49.1	45.5	83.5	78.8	53.0	73.1	84.2	86.5	87.5	**89.3**

just considering the average (last chart) it is obvious that the coder trained on COCO with augmentation performs generally better than the others, with some minor exceptions. Likewise, testing datasets differ appreciably from one another, for example ImageNet appears to be especially challenging under all conditions, followed by FFHQ. Finally, the training-testing alignment does not seem to impact significantly on the performance. For example, consider the charts corresponding to the LAION and COCO testing sets. The coders trained on COCO appear to perform better than those trained on LAION not only on the COCO testing set, but also on the LAION testing set, indicating that the quality of the training set prevails over the training-testing alignment.

We conclude with a note on the decision statistics. Even though the overall best combination seems to be $|\Delta^{01}|$ with the coder trained on COCO with augmentation, the Δ^{01} statistic (without modulus) ensures a very good performance uniformly over all trained encoders. In any case results seem to vary very much from one dataset to the other. For example D^0 works very well on RAISE and very badly on LAION. This suggests that a much deeper analysis is necessary and better decision statistics are probably yet to be found.

4 Effect of the dataset JPEG bias

Recent papers [4, 8] highlighted the presence of a bias in some datasets used in the field where all real images are JPEG compressed, while the generated images are stored in lossless format. Of course, if a detector is trained and tested on datasets affected by this bias, its results may be significantly altered. Our method is not affected by such distortion since it relies on a lossless encoder that provides an intrinsic model of real images which does not fit well to synthetic images. To confirm this fact, in Table 2 we report the same AUC results already shown in the main paper, but now averaged over each real dataset. At the top we have datasets with compressed images and at the bottom datasets with uncompressed images. Therefore, at the bottom we have a situation where both real and synthetic

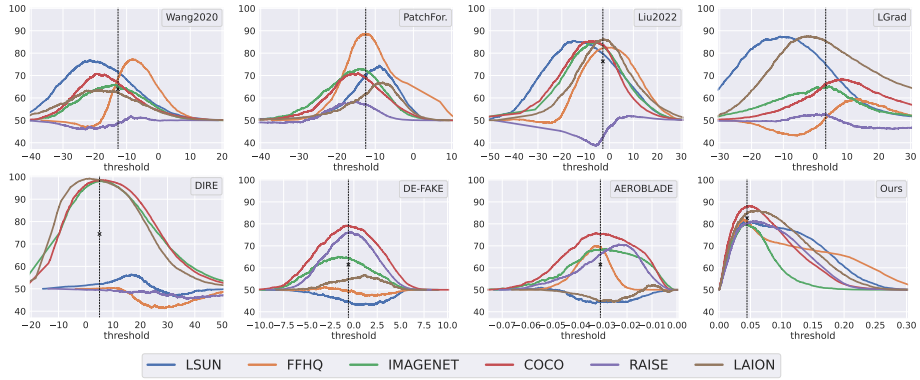

Fig. 2: Balanced accuracy as a function of the detection threshold for methods not analyzed in the main paper. For each dataset of real images, we average accuracy over all associated synthetic generators. The dotted vertical line indicates the global optimal threshold and the $\times$ symbol the corresponding accuracy. Again, only for the proposed method all peaks are very close, indicating the presence of a single threshold.

images are uncompressed and no JPEG-related distortion can affect the results. For our method there is no significant difference in performance between the top and bottom of the table. On the contrary, some SoTA methods show a more controversial behavior.

5 Additional comparison

In Figure 7 of the main paper, we report results in terms of balanced accuracy only for the proposed method and the best three competitors. For completeness, in Fig.2 we show the results of the other seven competitors compared again with those of the proposal. Like for the other competitors, the best threshold varies considerably depending on the dataset and hence no single threshold can provide good results in all cases.

References

1. Bammey, Q.: Synthbuster: Towards Detection of Diffusion Model Generated Images. IEEE Open Journal of Signal Processing (2023)
2. Chai, L., Bau, D., Lim, S.N., Isola, P.: What Makes Fake Images Detectable? Understanding Properties that Generalize. In: ECCV. pp. 103–120 (2020)
3. Corvi, R., Cozzolino, D., Zingarini, G., Poggi, G., Nagano, K., Verdoliva, L.: On the detection of synthetic images generated by diffusion models. In: ICASSP. pp. 1–5 (2023)
4. Grommelt, P., Weiss, L., Pfreundt, F.J., Keuper, J.: Fake or JPEG? Revealing Common Biases in Generated Image Detection Datasets. arXiv preprint arXiv:2403.17608 (2024)

5. Kang, M., Zhu, J.Y., Zhang, R., Park, J., Shechtman, E., Paris, S., Park, T.: Scaling up gans for text-to-image synthesis. In: CVPR. pp. 10124–10134 (2023)
6. Liu, B., Yang, F., Bi, X., Xiao, B., Li, W., Gao, X.: Detecting generated images by real images. In: ECCV. pp. 95–110 (2022)
7. Ojha, U., Li, Y., Lee, Y.J.: Towards universal fake image detectors that generalize across generative models. In: CVPR. pp. 24480–24489 (2023)
8. Ricker, J., Lukovnikov, D., Fischer, A.: AEROBLADE: Training-Free Detection of Latent Diffusion Images Using Autoencoder Reconstruction Error. In: CVPR. pp. 9130–9140 (2024)
9. Sha, Z., Li, Z., Yu, N., Zhang, Y.: DE-FAKE: Detection and Attribution of Fake Images Generated by Text-to-Image Generation Models. In: ACM SIGSAC. pp. 3418–3432 (2023)
10. Tan, C., Zhao, Y., Wei, S., Gu, G., Liu, P., Wei, Y.: Rethinking the Up-Sampling Operations in CNN-based Generative Network for Generalizable Deepfake Detection. In: CVPR. pp. 28130–28139 (2024)
11. Tan, C., Zhao, Y., Wei, S., Gu, G., Wei, Y.: Learning on Gradients: Generalized Artifacts Representation for GAN-Generated Images Detection. In: CVPR. pp. 12105–12114 (2023)
12. Wang, S.Y., Wang, O., Zhang, R., Owens, A., Efros, A.A.: CNN-generated images are surprisingly easy to spot... for now. In: CVPR. pp. 8692–8701 (2020)
13. Wang, Z., Bao, J., Zhou, W., Wang, W., Hu, H., Chen, H., Li, H.: DIRE for Diffusion-Generated Image Detection. ICCV pp. 22445–22455 (2023)