# A Bias-Free Training Paradigm for More General AI-generated Image Detection

## Supplementary Material

In this supplementary document, we report more details about our implementation (Sec. 8). Moreover, we briefly describe the state of the art methods we compare to (Sec. 9), and give more details about the calibration metrics used in the experiments (Sec. 10). We also provide additional ablation results in Sec. 11 and carry out further experiments on generalization (Sec. 12) on additional publicly available datasets. Furthermore, we provide more results on the robustness of our approach compared with SoTA methods (Sec. 13).

## 8. Implementation Details

**Training strategy.** The proposed model leverages the DINOv2+reg [7, 15] $504 \times 504$ image embedding network as its backbone, followed by a fully connected layer. The model is trained end-to-end using the binary cross-entropy loss function on an NVIDIA A100 GPU. The training process employs the ADAM optimizer with a learning rate of 1e-6, a weight decay of 1e-6, and a batch size of 24. During training, the balanced accuracy is evaluated on a validation set every 3435 iterations. Early stopping is applied to prevent overfitting: training is completed if the validation balanced accuracy does not improve by at least 0.1% over five consecutive evaluations.

**Test strategy.** If the test image is less than 504 pixels, padding is applied after patch embedding. Otherwise, we average the logit score over multiple crops to analyze the whole image.

**Training Dataset.** Here we give more deatils on how we built our dataset. Starting from the MS-COCO training set, consisting of 118K images with 80 categories of objects, we first discarded images with licenses different than Creative Commons. Before editing the images, we extracted the largest central crop, which allows us to retain most of the semantic content of the original image. We discarded images where objects are not present and ended up with a pristine source of 51,517 images. For content augmentation, we replaced the selected object with an object generated from the same category using the COCO segmentation mask, and from a different category using a rectangular box. We took care to not affect too much the realism of the content, so for the "different category" case the object is changed with one from a similar category, that belongs to the same COCO supercategory. In this scenario, the only exception is the category *person*, which does not have a supercategory and it is therefore replaced with a random object. As mentioned in the main paper, besides the default inpainting, we also consider a version where we take the pixels of the object from the generated image, and the pixels of the background from the original one. We did the same with the self-conditioned image, restoring the background with original pixels. Therefore, we ended up with six fake versions for each real image (Fig. 4 of the main paper).

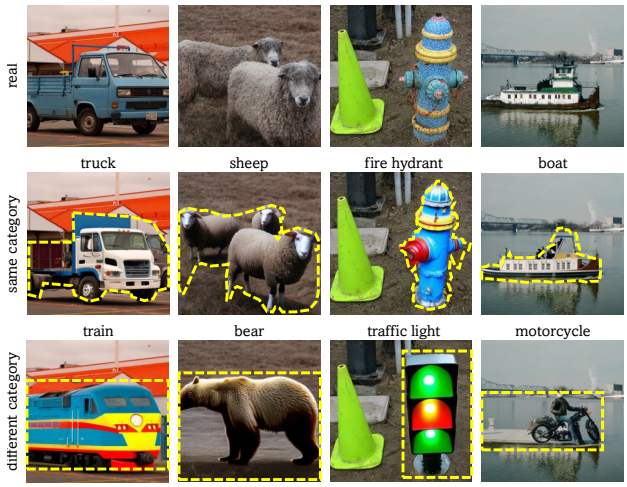


Figure 9. Examples of content augmented images from our training dataset. From real images (first row), we generate inpainted versions with the same content (second row) and different content (third row).

## 9. SoTA Methods

Below we provide a brief description of the methods we included in the comparison in Section 5 of our main paper. The training datasets used by these methods are indicated in Table 6 of the main paper.

**CNNDetect [20].** This is a CNN-based detector built on ResNet50 (pre-trained on ImageNet) that adpots augmentation in the form of post-processing operations, such as blurring and compression.

**DMID [6].** This work also relies on a ResNet-50, but it prevents down-sampling at the first layer so as to preserve the invisible forensics clues as much as possible, and uses a stronger augmentation to increase robustness.

**LGrad [17].** This work is also based on a ResNet-50 classifier, but this is fed by a generalized artifacts representation of the image in the form of gradients. This representation is designed to more effectively capture the artifacts introduced by synthetic generators.

| | Synthbuster | | | | | New Generators | | WildRF | | | AVG |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Training dataset | Midjourney | SDXL | DALL·E 2 | DALL·E 3 | Firefly | FLUX | SD 3.5 | Facebook | Reddit | Twitter | AUC↑/bAcc↑ |
| $D^3$ [2] | **99.8** / 85.3 | **100.** / **99.4** | **100.** / 87.5 | **100.** / 90.1 | **100.** / 89.7 | **98.8** / 63.1 | **99.9** / **96.4** | **98.3** / 88.8 | **95.4** / 86.7 | 98.1 / 88.2 | **99.0** / 87.5 |
| Ours | **99.9** / **98.8** | **100.** / **99.7** | **99.7** / **95.9** | **99.6** / **96.8** | **100.** / **99.6** | **97.9** / **85.3** | **99.3** / 95.1 | **98.0** / **95.0** | **96.0** / **89.8** | **99.4** / **96.5** | **99.0** / **95.2** |
| Training dataset | Midjourney | SDXL | DALL·E 2 | DALL·E 3 | Firefly | FLUX | SD 3.5 | Facebook | Reddit | Twitter | NLL↓/ECE↓ |
| $D^3$ [2] | 0.33 / .156 | 0.03 / .016 | 0.27 / .146 | 0.22 / .120 | 0.22 / .138 | 1.07 / .350 | **0.11** / .048 | 0.26 / .101 | 0.32 / .079 | 0.28 / .105 | 0.31 / .126 |
| Ours | **0.04** / **.008** | **0.01** / **.006** | **0.12** / **.044** | **0.10** / **.037** | **0.02** / **.011** | **0.40** / **.149** | 0.14 / **.044** | **0.17** / **.032** | **0.25** / **.043** | **0.11** / **.028** | **0.14** / **.040** |

Table 8. Ablation study on the influence of the training data. We compare the proposal with the same architecture DINOv2+reg trained on a publicly available dataset, $D^3$ [2], that includes 4 generators from the Stable Diffusion family. Performance are presented in terms of AUC/Accuracy (top) and ECE/NLL (bottom).

**UnivFD [14].** It exploits pre-trained CLIP features through linear probing. Fine-tuning is carried out on the same dataset of real and GAN-generated images as in [20].

**DeFake [16].** Both images and their corresponding prompts are used and fed into the visual and textual encoders of CLIP. The extracted features are the input of a multilayer perceptron trained for binary detection.

**DIRE [21].** It uses the reconstruction error of a generative model as the input of a ResNet-50. In fact, this error is expected to be lower for synthetic images than for real ones.

**AntifakePrompt [5].** It relies on a visual question-answering (VQA) tool, InstructBLIP. The VQA is used with a fixed question, "Is this photo real?", and fine-tuned to provide accurate responses ("Yes" or "No") using a soft prompt tuning technique. Note that the method provides hard binary predictions hence only accuracy can be computed.

**NPR [18].** In this case a ResNet-50 is fed using a residual image computed as the difference between the original image and its interpolated version. The idea is to exploit the artifacts related to the up-sampling process which is common in several generative models.

**FatFormer [13].** It adopts CLIP and introduces forgery-aware adapters to extract forensic traces from both space and frequency domains. The method proposes a language-guided alignment mechanism to supervise the process and ensure the association between image and text.

**FasterThanLies [9].** The method employs a Binary Neural Network for features extraction phase and a linear classifier for detection. Beyond the image, the model has two additional input channels: the Fast Fourier Transform magnitude and the Local Binary Pattern image. We report results using the unfrozen BNext-M backbone.

**RINE [8].** It uses features extracted from the intermediate blocks of a CLIP encoder and an additional trainable module to take into account the influence of each block on the final decision.

**AIDE [22].** It leverages hybrid features extracted from a ConvNeXt-based Open CLIP model and a CNN which is fed with patches filtered to remove semantic content and exploit low-level artifacts.

**LaDeDA [3].** It is a patch-based classifier that leverages local image features. The image is split into multiple patches, for each patch a prediction is computed and then averaged to obtain the image-level prediction.

**C2P-CLIP [19].** It uses the Low-Rank Adaptation (LoRA) strategy to fine-tune the image encoder of CLIP. Moreover, it relies on a contrastive learning strategy based on category prompts.

**CoDE [2].** CoDE trains a Vision Transformer using a contrastive loss similar to CLIP. However, while CLIP aims to learn features for text-image matching, CoDE aims at obtaining an embedding space where real and fake images are effectively separated. We report results using CoDE in combination with the linear classifier.

## 10. Calibration Metrics

Here we provide some more details about the calibration metrics used in the paper. The binary Expected Calibration Error (ECE) is defined as:

$$\mathrm{ECE} = \sum_{m=1}^{M} \frac{|B_m|}{N} \left|\mathrm{prob}(B_m) - \mathrm{pred}(B_m)\right| \qquad (1)$$

where $N$ is the number of samples of the test-set, $M$ is the number of bins, and $B_m$ is the set of samples whose predictions fall into the $m$-th bin, with $|B_m|$ its cardinality. $\mathrm{prob}(B_m)$ and $\mathrm{pred}(B_m)$ are the actual probability and the average predicted probability of the target class in that bin, respectively. In case of unbalanced test-set, we weigh the contribution of each sample in the average to re-balance the relevance between two classes. We used $M = 15$ bins.

The balanced Negative Log-Likelihood (NLL) is defined as:

$$\mathrm{NLL} = -\frac{0.5}{|S_0|} \sum_{i \in S_0} \log p_i(0) - \frac{0.5}{|S_1|} \sum_{i \in S_1} \log p_i(1) \qquad (2)$$

where $S_0$ and $S_1$ are the set of samples of non-target and target class, respectively, while $p_i(0)$ and $p_i(1)$ are the pre-

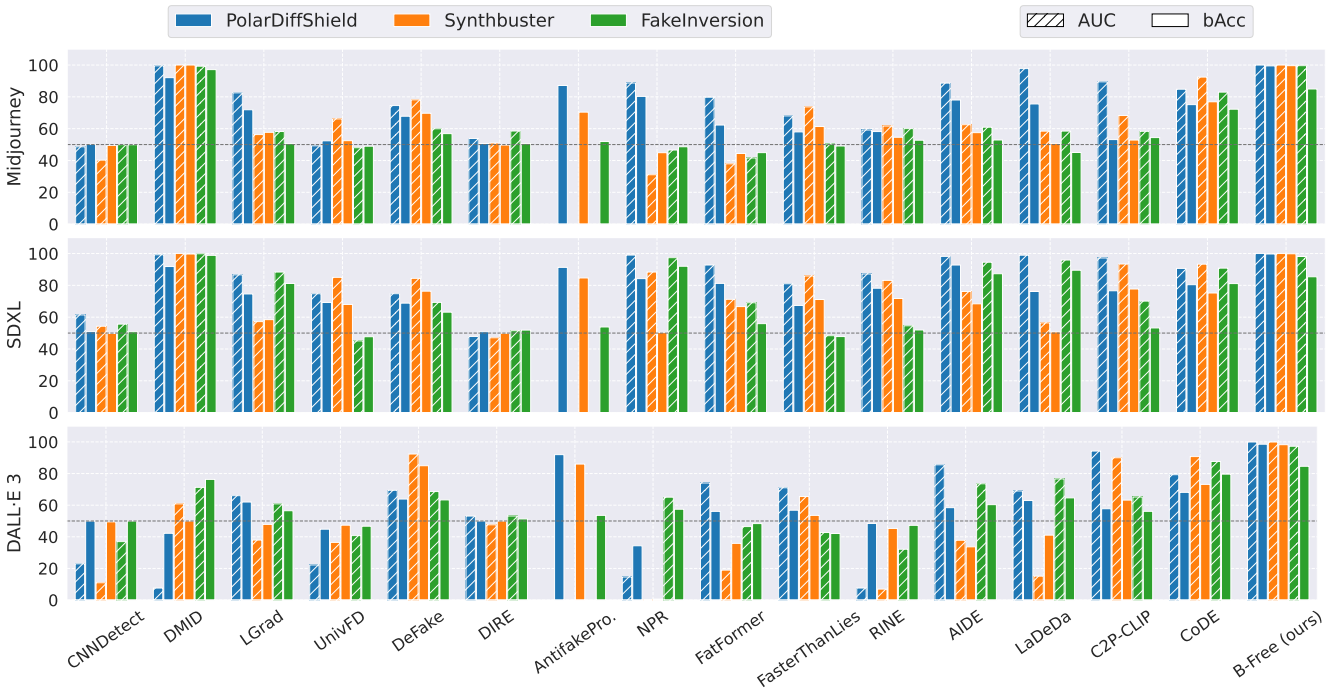


Figure 10. SoTA performance evaluated in terms of AUC and balanced Accuracy on Midjourney, SDXL and DALL·E generators from different datasets.

dicted probabilities of the two classes for the $i$-th input sample.

## 11. Additional Ablation

In this Section we further investigate the influence of our training dataset. We train our network on the very recent Diffusion-generated Deepfake Detection ($D^3$) dataset [2], of about 8M synthetic images (from 256 $\times$ 256 to 1024 $\times$ 1024) from the generators SD 1.4, SD 2.1, SDXL, and DeepFloyd IF. The images are generated using prompts taken from the description of the real source from LAION (text-driven generation). From Tab. 8, we can notice that although the AUC is similar, there is a significant increase in terms of balanced accuracy (87.5 vs 95.2) and decrease in terms of both NLL (0.31 vs 0.14) and ECE (0.13 vs 0.04). This confirms that our training paradigm enables better calibration and improved generalization.

## 12. Additional Generalization Analysis

In this Section we conduct further experiments which confirm that our method can generalize better than other methods and obtain less biased results. We also detail results and show the performance on each synthetic generator for GenImage and FakeBench datasets.

**Evaluation on same generators from different datasets.** Here we further expand on the analysis conducted in Fig. 2 of the main paper where we have shown that some detectors achieve different performance on the same generator when the images are taken from two different datasets. This puzzling behavior suggests the possibility that these methods rely on subtle dataset biases besides true traces left by the synthetic generator. In Fig. 10 we extend this analysis to all SoTA methods described in Sec. 9. More specifically, we analyze the performance in terms of AUC and balanced accuracy over three synthetic generators: Midjourney, SDXL and DALL-E 3 that come from three different datasets PolarDiffShield [11], Synthbuster [1] and FakeInversion [4]. As said before, for several methods the performance is not consistent on the same generator and can vary even by 20% from one dataset to another. In addition, for some methods the AUC is around 50%, which corresponds to random choice, or even below 50% which means that the detector tends to invert the labels between real and fake.

**Evaluation on different synthetic generators.** We conduct a more detailed analysis of the results on GenImage (unbiased), where fake images have been subjected to JPEG compression, similar to real images, to prevent detectors from exploiting compression artifacts. We also consider FakeBench [12], that consists of 3,000 real and 3,000 fake images generated by 10 different models. These datasets include both GAN and Diffusion-based synthetic images, which allows us to better understand the ability of our approach to generalize to different architectures. Results are presented in Tab. 9 and Tab. 10. We note that our approach obtains very good results consistently across almost

| AUC↑/bAcc↑ | GenImage (unbiased) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | BigGAN | VQDM | ADM | GLIDE | SD 1.4 | SD 1.5 | Midjourney | Wukong | AVG |
| CNNDetect | 70.9 / 58.4 | 63.4 / 51.2 | 51.8 / 49.9 | 59.4 / 50.7 | 65.1 / 50.1 | 66.4 / 49.9 | 79.3 / 50.1 | 62.6 / 50.2 | 64.8 / 51.3 |
| DMID | 74.6 / 52.3 | **97.6** / 75.1 | 78.5 / 51.3 | 94.9 / 56.6 | **100. / 99.9** | **100. / 99.8** | **100. / 97.4** | **100. / 99.6** | 93.2 / 79.0 |
| LGrad | 18.7 / 28.9 | 23.9 / 30.8 | 24.6 / 30.5 | 22.2 / 30.0 | 50.0 / 49.8 | 49.2 / 49.1 | 50.5 / 50.6 | 47.6 / 46.9 | 35.8 / 39.6 |
| UnivFD | 96.7 / 86.1 | 94.8 / 79.7 | 85.2 / 64.4 | 88.8 / 63.9 | 78.7 / 55.5 | 78.1 / 56.6 | 74.0 / 54.2 | 86.9 / 63.7 | 85.4 / 65.5 |
| DeFake | 72.6 / 64.4 | 71.1 / 64.4 | 49.3 / 48.5 | 87.9 / 80.4 | 93.3 / 85.1 | 93.4 / 85.4 | 87.7 / 79.2 | 89.8 / 81.8 | 80.6 / 73.7 |
| DIRE | 26.6 / 46.9 | 35.0 / 47.7 | 25.3 / 46.7 | 29.9 / 47.0 | 41.7 / 47.3 | 39.8 / 47.3 | 38.0 / 47.5 | 45.4 / 47.7 | 35.2 / 47.3 |
| AntifakePrompt | - / 81.7 | - / 81.1 | - / **81.6** | - / 81.8 | - / 77.1 | - / 76.6 | - / 70.4 | - / 77.6 | - / 78.5 |
| NPR | 56.9 / 56.3 | 52.3 / 53.9 | 46.9 / 50.5 | 42.1 / 48.3 | 54.3 / 49.4 | 53.3 / 49.7 | 42.3 / 47.4 | 52.4 / 50.2 | 50.1 / 50.7 |
| FatFormer | 88.5 / 80.1 | 84.5 / 71.5 | 69.1 / 60.4 | 78.4 / 65.1 | 49.8 / 52.0 | 48.7 / 53.3 | 46.2 / 51.6 | 61.6 / 58.1 | 65.9 / 61.5 |
| FasterThanLies | 78.9 / 54.1 | 86.8 / 76.6 | 88.6 / 77.2 | 83.0 / 66.1 | 97.8 / 92.2 | 97.9 / 92.3 | 83.1 / 69.7 | 95.4 / 88.1 | 88.9 / 77.0 |
| RINE | **99.4 / 88.5** | **98.4** / 81.4 | **93.8** / 63.9 | **98.1** / 74.7 | 93.9 / 60.5 | 94.1 / 61.1 | 86.3 / 52.4 | 95.7 / 70.0 | 95.0 / 69.1 |
| AIDE | 73.1 / 50.7 | 78.0 / 51.0 | 61.2 / 50.1 | 80.4 / 52.3 | 98.2 / 74.5 | 98.5 / 75.9 | 88.1 / 57.4 | 95.9 / 69.3 | 84.2 / 60.2 |
| LaDeDa | 93.1 / 80.3 | 10.8 / 34.8 | 6.8 / 34.6 | 8.8 / 34.5 | 55.6 / 54.8 | 53.6 / 53.0 | 51.3 / 52.1 | 61.6 / 57.7 | 42.7 / 50.2 |
| C2P-CLIP | 97.2 / 87.5 | 92.2 / 74.1 | 86.7 / 71.3 | 93.6 / 74.8 | 94.4 / 80.5 | 94.3 / 79.1 | 76.3 / 55.9 | 93.1 / 81.0 | 91.0 / 75.5 |
| CoDE | 70.2 / 50.0 | 66.8 / 56.0 | 53.7 / 51.9 | 78.1 / 58.0 | **99.4** / 96.6 | **99.2** / 96.5 | 86.0 / 69.6 | **99.1** / 95.0 | 81.6 / 71.7 |
| B-Free (ours) | 94.1 / 68.7 | 97.0 / **88.7** | **93.0** / 79.8 | 95.8 / **85.3** | **100.** / 98.8 | **100.** / 98.8 | **99.2** / 95.7 | **100. / 99.0** | **97.4 / 89.3** |

Table 9. Performance on each generator included in GenImage (unbiased) dataset in terms of AUC and balanced Accuracy. Bold underlines the best performance for each column with a margin of 1%.

| AUC↑/bAcc↑ | FakeBench | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | ProGAN | StyleGAN | FuseDream | VQDM | GLIDE | CogView2 | DALL·E 2 | DALL·E 3 | SD | Midjourney | AVG |
| CNNDetect | **100. / 99.7** | 98.3 / 75.1 | 94.8 / 61.1 | 62.9 / 51.9 | 62.6 / 50.6 | 64.9 / 49.7 | 56.1 / 49.7 | 58.6 / 49.7 | 57.2 / 49.7 | 62.0 / 49.9 | 71.7 / 58.7 |
| DMID | 61.0 / 51.1 | 80.1 / 52.1 | 93.1 / 52.4 | 97.8 / 79.7 | 94.0 / 63.2 | **100. / 99.7** | 94.9 / 55.1 | 96.7 / 88.9 | **100. / 99.1** | **97.3** / 90.7 | 91.5 / 73.2 |
| LGrad | 96.8 / 77.1 | 82.3 / 72.9 | 18.9 / 28.4 | 75.2 / 68.6 | 41.8 / 43.9 | 23.7 / 33.6 | 10.9 / 27.6 | 30.6 / 35.6 | 24.7 / 34.1 | 76.1 / 67.3 | 48.1 / 48.9 |
| UnivFD | **99.9** / 98.6 | 96.0 / 83.4 | **99.2 / 96.3** | 94.6 / 77.3 | 86.5 / 62.8 | 84.7 / 63.1 | 88.0 / 65.9 | 69.6 / 55.8 | 76.8 / 56.4 | 65.5 / 55.6 | 86.1 / 71.5 |
| DeFake | 63.7 / 58.1 | 73.7 / 66.7 | 53.8 / 51.0 | 69.8 / 64.5 | 81.6 / 74.2 | 84.7 / 77.2 | 83.6 / 76.5 | 81.7 / 74.5 | 86.4 / 77.3 | 78.7 / 70.5 | 75.8 / 69.0 |
| DIRE | 90.4 / 89.5 | 56.6 / 55.4 | 23.7 / 40.0 | 91.3 / 89.2 | 53.2 / 63.7 | 36.7 / 41.0 | 44.2 / 43.0 | 76.6 / 74.5 | 47.7 / 49.7 | 83.4 / 81.2 | 60.4 / 62.7 |
| AntifakePrompt | - / 79.0 | - / 78.0 | - / 78.6 | - / 77.0 | - / 78.8 | - / 75.8 | - / 73.4 | - / 74.0 | - / 71.6 | - / 76.1 | - / 76.2 |
| NPR | **99.5** / 92.4 | 78.1 / 68.1 | 48.7 / 42.7 | 93.4 / **90.9** | 67.0 / 65.2 | 50.3 / 42.7 | 41.7 / 42.9 | 46.3 / 44.6 | 57.5 / 51.4 | 89.0 / 84.6 | 67.1 / 62.5 |
| FatFormer | **100.** / 97.6 | **99.3 / 97.1** | 90.7 / 81.8 | 96.8 / 88.5 | 74.2 / 69.0 | 47.1 / 53.3 | 45.3 / 48.1 | 52.4 / 49.5 | 50.4 / 51.0 | 79.6 / 64.6 | 73.6 / 70.0 |
| FasterThanLies | 87.0 / 80.2 | 72.4 / 57.7 | 85.7 / 75.4 | 54.7 / 45.9 | 76.9 / 62.5 | 96.0 / 87.7 | 92.9 / 85.7 | 73.6 / 60.6 | 93.6 / 84.5 | 67.6 / 57.7 | 80.0 / 69.8 |
| RINE | **100. / 99.6** | **99.3** / 95.1 | **99.8 / 96.6** | **98.8** / 88.6 | **95.4** / 70.2 | 86.7 / 59.2 | 93.0 / 60.9 | 75.1 / 52.6 | 85.5 / 55.9 | 82.2 / 61.1 | 91.6 / 74.0 |
| AIDE | 89.4 / 64.3 | 89.4 / 70.0 | 71.7 / 47.3 | 90.7 / 78.1 | 79.7 / 68.3 | 85.5 / 60.0 | 84.1 / 52.6 | 88.0 / 61.9 | 86.0 / 64.6 | 88.0 / 71.5 | 85.2 / 63.9 |
| LaDeDa | 98.0 / 82.5 | 94.5 / 82.5 | 37.2 / 40.5 | 85.8 / 81.5 | 52.6 / 57.3 | 39.4 / 41.6 | 36.4 / 35.3 | 49.9 / 45.1 | 45.3 / 46.1 | 90.7 / 78.1 | 63.0 / 59.1 |
| C2P-CLIP | **100. / 99.5** | **99.4 / 98.0** | 98.2 / 93.0 | 97.1 / 86.7 | 91.9 / 76.4 | 67.3 / 61.7 | 72.6 / 56.9 | 74.7 / 55.5 | 74.9 / 59.9 | 88.1 / 58.0 | 86.4 / 74.6 |
| CoDE | 64.3 / 52.5 | 53.0 / 49.5 | 73.4 / 56.3 | 78.4 / 61.7 | 91.6 / 78.0 | 97.7 / 93.7 | 93.8 / 82.8 | 95.8 / 89.2 | **99.5** / 96.2 | 89.7 / 76.7 | 83.7 / 73.7 |
| B-Free (ours) | **99.3** / 96.4 | 97.7 / 88.5 | **99.3** / 95.2 | 96.5 / 86.5 | **95.5 / 87.7** | **100. / 98.9** | **98.7 / 94.9** | **99.9 / 98.7** | **100. / 98.7** | **98.0 / 94.5** | **98.5 / 94.0** |

Table 10. Performance on each generator included in FakeBench dataset in terms of AUC and balanced Accuracy. Bold underlines the best performance for each column with a margin of 1%.

all generators, while other methods, such as DMID, UnivFD, RINE, FasterThanLies, and FatFormer, perform very well in terms of AUC only on certain generators. In addition, for our method the gap between AUC and balanced accuracy is reduced which ensures more reliable results.

Finally, we generated a set of 2,000 images with two autoregressive models [10, 23], whose architecture vastly differs from the training data. On both models we obtain good results with an average of 99% of AUC and 94.2% of accuracy, probably due to the similarity between the tokenizer used in these models and the latent embedders of Stable Diffusion models.

## 13. Additional Robustness Analysis

We repeat the experiment reported in Figure 6 to test robustness under various operations and carry out comparisons with SOTA methods. Results are shown in Fig. 11 under three post-processing operations: JPEG compression, resizing, and blurring. We can observe that our approach is more robust by a large margin compared with other methods and can ensure a balanced accuracy that is always above 80% even in the most challenging scenario.

Finally we further investigate robustness performance on FakeInversion [4], where the real images have been retrieved from the web. To better understand the effect of compression and resizing we compare the performance when applying such operations. In particular, to simulate the upload on social networks, we resize with a scale factor randomly sampled between 0.7 and 1, and compress with a JPEG quality factor between 70 and 100. In Tab. 11 results show that the performance on such dataset drops substantially, except for DMID and our method, though our approach has better calibration metrics.

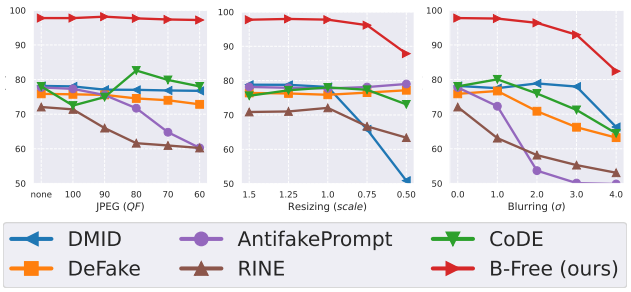


Figure 11. Robustness analysis in terms of balanced Accuracy carried out on nine generators of Synthbuster under three different post-processing operations: JPEG compression, resizing and blurring. The five best performing SoTA detectors on Synthbuster have been included in the analysis.

| | Original | | Social network simulation | |
|---|---|---|---|---|
| | AUC↑/bAcc↑ | NLL↓/ECE↓ | AUC↑/bAcc↑ | NLL↓/ECE↓ |
| CNNDetect | 54.3 / 50.9 | 7.94 / .488 | 51.8 / 50.1 | 8.73 / .498 |
| DMID | 97.3 / **96.1** | **0.25 / .041** | 94.4 / 81.3 | 0.55 / .182 |
| LGrad | 84.4 / 77.2 | 2.27 / .200 | 60.2 / 55.4 | 7.59 / .426 |
| UnivFD | 54.9 / 52.8 | 2.19 / .391 | 49.7 / 50.0 | 2.57 / .434 |
| DeFake | 69.8 / 63.3 | 0.95 / .225 | 69.3 / 62.7 | 0.98 / .236 |
| DIRE | 53.3 / 51.8 | 13.4 / .360 | 55.5 / 51.7 | 13.4 / .358 |
| AntifakePrompt | - / 53.9 | - / - | - / 54.8 | - / - |
| NPR | 91.6 / 87.0 | 4.96 / .123 | 43.3 / 49.9 | 27.1 / .501 |
| FatFormer | 68.2 / 59.7 | 3.45 / .386 | 48.7 / 50.4 | 5.44 / .490 |
| FasterThanLies | 49.7 / 48.6 | 3.64 / .476 | 51.0 / 49.9 | 3.13 / .458 |
| RINE | 69.6 / 63.6 | 4.84 / .319 | 62.4 / 52.9 | 6.28 / .437 |
| AIDE | 85.5 / 76.9 | 0.54 / .137 | 67.3 / 56.4 | 0.93 / .276 |
| LaDeDa | 91.7 / 84.7 | 3.03 / .129 | 51.9 / 53.1 | 24.8 / .454 |
| C2P-CLIP | 74.1 / 59.6 | 0.82 / .260 | 71.9 / 59.0 | 0.89 / .284 |
| CoDE | 87.5 / 78.7 | 0.74 / .143 | 82.5 / 74.4 | 0.89 / .173 |
| B-Free (ours) | **99.3** / 86.3 | 0.32 / .144 | **98.5 / 86.4** | **0.32 / .131** |

Table 11. Performance on FakeInversion dataset. We show results on the original dataset and on a post-processed version, to simulate the upload on social networks.

## References


[1] Quentin Bammey. Synthbuster: Towards detection of diffusion model generated images. *IEEE Open Journal of Signal Processing*, 2023. 3

[2] Lorenzo Baraldi, Federico Cocchi, Marcella Cornia, Lorenzo Baraldi, Alessandro Nicolosi, and Rita Cucchiara. Contrasting Deepfakes Diffusion via Contrastive Learning and Global-Local Similarities. In *ECCV*, 2024. 2, 3

[3] Bar Cavia, Eliahu Horwitz, Tal Reiss, and Yedid Hoshen. Real-Time Deepfake Detection in the Real-World. *arXiv preprint arXiv:2406.09398*, 2024. 2

[4] George Cazenavette, Avneesh Sud, Thomas Leung, and Ben Usman. FakeInversion: Learning to Detect Images from Unseen Text-to-Image Models by Inverting Stable Diffusion. In *CVPR*, pages 10759–10769, 2024. 3, 4

[5] You-Ming Chang, Chen Yeh, Wei-Chen Chiu, and Ning Yu. AntifakePrompt: Prompt-Tuned Vision-Language Models are Fake Image Detectors. *arXiv preprint arXiv:2310.17419*, 2023. 2

[6] Riccardo Corvi, Davide Cozzolino, Giada Zingarini, Giovanni Poggi, Koki Nagano, and Luisa Verdoliva. On the detection of synthetic images generated by diffusion models. In *ICASSP*, pages 1–5, 2023. 1

[7] Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision Transformers Need Registers. In *ICLR*, 2024. 1

[8] Christos Koutlis and Symeon Papadopoulos. Leveraging Representations from Intermediate Encoder-blocks for Synthetic Image Detection. In *ECCV*, pages 394–411, 2024. 2

[9] Romeo Lanzino, Federico Fontana, Anxhelo Diko, Marco Raoul Marini, and Luigi Cinque. Faster Than Lies: Real-time Deepfake Detection using Binary Neural Networks. In *CVPR Workshops*, pages 3771–3780, 2024. 2

[10] Tianhong Li, Yonglong Tian, He Li, Mingyang Deng, and Kaiming He. Autoregressive Image Generation without Vector Quantization. In *NeurIPS*, pages 56424–56445, 2024. 4

[11] Yanhao Li, Quentin Bammey, Marina Gardella, Tina Nikoukhah, Jean-Michel Morel, Miguel Colom, and Rafael Grompone Von Gioi. MaskSim: Detection of Synthetic Images by Masked Spectrum Similarity Analysis. In *CVPR*, pages 3855–3865, 2024. 3

[12] Yixuan Li, Xuelin Liu, Xiaoyang Wang, Bu Sung Lee, Shiqi Wang, Anderson Rocha, and Weisi Lin. FakeBench: Probing Explainable Fake Image Detection via Large Multimodal Models. *arXiv preprint arXiv:2404.13306*, 2024. 3

[13] Huan Liu, Zichang Tan, Chuangchuang Tan, Yunchao Wei, Jingdong Wang, and Yao Zhao. Forgery-aware Adaptive Transformer for Generalizable Synthetic Image Detection. In *CVPR*, pages 10770–10780, 2024. 2

[14] Utkarsh Ojha, Yuheng Li, and Yong Jae Lee. Towards universal fake image detectors that generalize across generative models. In *CVPR*, pages 24480–24489, 2023. 2

[15] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. DINOv2: Learning Robust Visual Features without Supervision. *Transactions on Machine Learning Research Journal*, 2024. 1

[16] Zeyang Sha, Zheng Li, Ning Yu, and Yang Zhang. DE-FAKE: Detection and Attribution of Fake Images Generated by Text-to-Image Generation Models. In *ACM SIGSAC*, pages 3418–3432, 2023. 2

[17] Chuangchuang Tan, Yao Zhao, Shikui Wei, Guanghua Gu, and Yunchao Wei. Learning on Gradients: Generalized Artifacts Representation for GAN-Generated Images Detection. In *CVPR*, pages 12105–12114, 2023. 1

[18] Chuangchuang Tan, Huan Liu, Yao Zhao, Shikui Wei, Guanghua Gu, Ping Liu, and Yunchao Wei. Rethinking the Up-Sampling Operations in CNN-based Generative Network for Generalizable Deepfake Detection. In *CVPR*, 2024. 2

[19] Chuangchuang Tan, Renshuai Tao, Huan Liu, Guanghua Gu, Baoyuan Wu, Yao Zhao, and Yunchao Wei. C2P-CLIP: Injecting Category Common Prompt in CLIP to Enhance Generalization in Deepfake Detection. *arXiv preprint arXiv:2408.09647*, 2024. 2

[20] Sheng-Yu Wang, Oliver Wang, Richard Zhang, Andrew Owens, and Alexei A Efros. CNN-generated images are surprisingly easy to spot... for now. In *CVPR*, pages 8695–8704, 2020. 1, 2

[21] Zhendong Wang, Jianmin Bao, Wengang Zhou, Weilun Wang, Hezhen Hu, Hong Chen, and Houqiang Li. DIRE for diffusion-generated image detection. In *ICCV*, pages 22445–22455, 2023. 2

[22] Shilin Yan, Ouxiang Li, Jiayin Cai, Yanbin Hao, Xiaolong Jiang, Yao Hu, and Weidi Xie. A Sanity Check for AI-generated Image Detection. In *ICLR*, 2025. 2

[23] Qihang Yu, Ju He, Xueqing Deng, Xiaohui Shen, and Liang-Chieh Chen. Randomized Autoregressive Visual Generation. *arXiv preprint arXiv:2411.00776*, 2024. 4