jet INTL JOURNAL OF ELECTRONICS AND TELECOMMUNICATIONS, 2026, VOL. 72, NO. 2, PP. 1–7
Manuscript received October 29, 2026; revised April 2026. doi: 10.24425/ijet.2026.157932

AI-generated image detection using Machine Learning techniques

Jan Chorąży, and Łukasz Dąbała

***Streszczenie*—Rapid developments in the field of generative artificial intelligence have enabled the creation of photorealistic images that are becoming increasingly difficult to distinguish from real photographs. This work aims to implement or adapt three contemporary methods for generated image detection: a Photo-Response Non-uniformity (PRNU) extractor paired with a custom CNN, an Error-Level Analysis (ELA) extractor coupled with the same custom CNN, and an adaptation of the Latent Reconstruction Error framework LaRE^2. Each of the methods has been thoroughly tested on a custom dataset, which was constructed by combining part of the Tiny GenImage dataset with images generated by a state-of-the-art transformer-diffusion model named FLUX. The data, balanced evenly between real and fake images, has been separated into five subsets, each respective to one of the included models: ADM, BigGAN, FLUX, Midjourney, and Stable Diffusion v1.5. Experiments were conducted in three different categories, to ensure proper validation of performance of the tested methods.**

***Keywords*—image generation, image recognition, generative AI**

I. Introduction

RECENT breakthroughs in the field of generative AI have enabled the creation of photorealistic images, which are becoming increasingly difficult to detect. Generative Adversarial Networks (GANs) were the first to garner attention in the field, but due to their relatively simple nature, detection systems never struggled against these architectures. More recently, diffusion models such as Stable Diffusion have started posing a considerable threat to detection systems. The biggest problem in the area of generated image detection is creating a model that will perform well regardless of the investigated generational model. To this end, researchers have elected to leverage deeper representations of image structures and latent features.

The main contribution of this study is the extensive evaluation of three novel approaches to generated image detection: a Photo-Response Non-uniformity (PRNU) extractor paired with a custom CNN, an Error-Level Analysis (ELA) extractor coupled with the same custom CNN, and an adaptation of the Latent Reconstruction Error framework LaRE^2, all on an enhanced version of an established dataset that includes transformer-based diffusion generation. The second one is the creation of the aforementioned custom dataset by joining parts of the GenImage dataset with newly generated images from the FLUX model.

All authors are with Warsaw University of Technology, Poland (e-mail: jan.chorazy02@gmail.com, Lukasz.Dabala@pw.edu.pl)

II. Related Work

A. Advances in AI-based Image Generation

In recent years, interest in the field of generative AI has drastically increased, leading to numerous breakthrough developments. The first notable networks were Generative Adversarial Networks (GANs) [1], introduced by Goodfellow in 2014. GANs use a dual framework wherein a generator tries to create an image based on a given class or prompt, while a discriminator learns to distinguish between real and fake images. Among numerous notable variants, BigGAN [7] represents a significant improvement both in the quality of the generation and class diversity by utilizing large-scale learning on the ImageNet [15] dataset as well as improved architecture. More recently, diffusion models emerged as a powerful architecture that creates images through an iterative process of denoising an initial Gaussian noise distribution. Notable examples include Augmented Diffusion Models (ADMs) and Stable Diffusion v1.5 [2], which improve computational efficiency and generation speed by leveraging latent space optimization.

B. Challenges in detecting AI images

The increasing quality of generated images poses a significant challenge to established detection techniques. As the study [21] shows, humans are not so skilled in fake image recognition - only 62% of images were correctly classified. A major challenge of synthetic detection is a lack of consistent artifacts between different generational models. Each distinct architecture introduces different statistical properties into the image it produces, with factors such as loss function or dataset further diversifying the artifacts. As such, the effectiveness of contemporary detectors is greatly affected, with the result being a lack of universal indicators for detecting generated images, limiting the effectiveness of general-purpose classifiers. What is more, nearly all tests are conducted using only benchmarks, which greatly improves the result. In case of artifacts, compression, or some common filters are introduced, the accuracy of the detection is greatly reduced [22].

C. Comparative studies and benchmarks

Several novel studies have been conducted surrounding the topic of generalizability of fake image detection systems. Papers such as [18], [16] and [17] focus on benchmarking available methods, while others such as [19] and [20] propose new solutions that improve over previous models. These works reveal—as often explicitly stated by the authors—that many detectors, perform well when trained and tested on images from the same model but fail to generalize on cross-model datasets, resulting in poor performance, underlining the need for more complex investigation strategies.

III. Investigated Methods

We implemented three novel methods of synthetic image detection, each based on a different approach to feature extraction.

A. Photo-Response Non-Uniformity (PRNU)

1) Theoretical Background: Photo-Response Non-Uniformity (PRNU) [3] [5] is a technique initially introduced to find the source device responsible for taking a given picture. As observed by Lukas in [5], PRNU is part of the sensor pattern noise, as well as its most stable component. It is the cause of manufacturing imperfections of a digital camera's photodiodes, and as such it is random and unique to every single unit. Since PRNU patterns are dependent solely on the physical image sensor, they do not change over time (unless the image sensor deteriorates). Taking into account the mentioned qualities, PRNU can be considered a sort of "fingerprint"for the camera.

2) PRNU extraction: The PRNU pattern can be mathematically extracted from an image, which is modeled by equation:

$$Img_{out} = (I_{ones} + Noise_{PRNU}) \cdot Img_{in} + Noise_{add}$$

Where Img_{out} is the image captured by the camera, I_{ones} is a matrix full of ones, $Noise_{PRNU}$ is the PRNU pattern, and Img_{in} is the 'ideal' image without disturbance, as presented to the camera. The symbol $\cdot$ denotes a matrix dot product and $Noise_{add}$ represents the additive noise from all other sources. Traditionally, extraction of the PRNU pattern involves taking the average noise residual value obtained from multiple photos taken by the same camera. However, for the application of AI image detection, each image has to be considered individually. The extraction process first involves computing the noise residual using the following equation:

$$W = Img_{out} - denoise(Img_{out}) \tag{1}$$

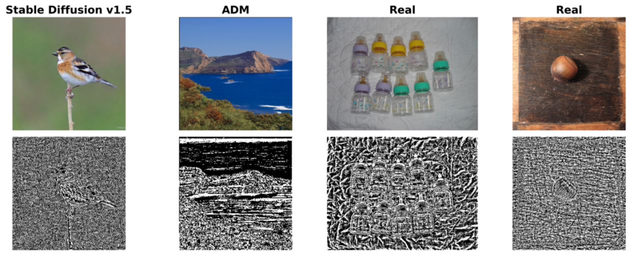
Where Img_{out} is the image generated by a camera, W is the noise residual and $denoise()$ represents the denoising process. We assume that the additive noise can be neglected and that $Img_{in} = denoise(Img_{out})$. Equation 2 is used to compute the PRNU pattern for N images coming from the same camera, which for us will always be $N = 1$.

$$F = \frac{\sum_{n=1}^{N} W^n Img_{in}^n}{\sum_{n=1}^{N} (Img_{in}^n)^2} \tag{2}$$

Which, for a single image($N = 1$) can be reduced to:

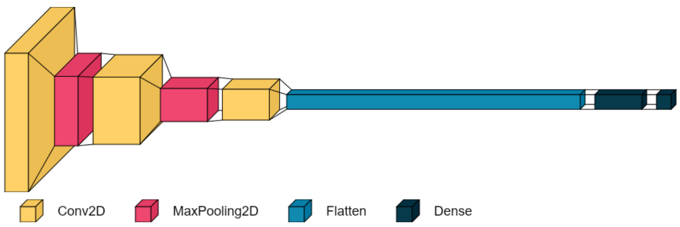
$$F = \frac{W}{Img_{in}} \tag{3}$$

Figure 1 shows examples of images before and after PRNU pattern extraction.



Rysunek 1. Upper row: original image; bottom row: PRNU pattern

3) Use in AI detection: AI-generated images do not come from any physical medium and therefore cannot exhibit authentic PRNU patterns. Some generators might be able to produce texture-like artifacts or inject noise into the output; however, these additions are not consistent, sensor-driven patterns, like the ones observed in real images.
In this work, PRNU is used as a feature extractor for a custom CNN. Each input image is processed to extract the noise residual using a denoising algorithm (in this case, the non-local means denoising algorithm). The resulting PRNU map is treated as a grayscale input into the network. A custom CNN architecture, visualized in figure 2, is trained to distinguish the differences in noise patterns for real and generated images and to perform binary classification.



Rysunek 2. Visualization of the custom CNN architecture

B. Error Level Analysis (ELA)

1) Theoretical Background: Error Level Analysis (ELA) [3] [4] is a forensic analysis technique originally used to detect localized modifications in digital images by leveraging the characteristics of lossy compression, in particular the JPEG format. As proposed by Martin-Rodriguez in [3], ELA can be adopted to suit the task of distinguishing between real and generated images. In particular, it is used as a pre-processing step, while a custom CNN works as the actual classifier.

2) ELA extraction: ELA takes advantage of the fact that images that have gone through several rounds of compression - especially those with edits or synthetic content inserted - tend to exhibit **non-uniform error levels** in different regions. ELA extraction is done in the following steps:

1) **Initial compression:** The investigated image is compressed by a fixed high compression rate (e.g., 95%)
2) **Decompression:** The obtained compressed version is decoded.
3) **Difference computation:**The modified and original versions of the image undergo pixel-wise difference computation.

The result is an ELA map of the image, where areas where the compression introduced significant artifacts are highlighted, as shown in Figure 3. The highlights often correspond to areas with denser spatial content or modifications. In theory, real images should result in ELA maps with spatial disparity, corresponding to the natural content distribution on the image. Generated images, however, should show abnormally smooth or globally uniform error patterns due to the lack of initial compression or proper frequency distribution.

Rysunek 3. Upper row: original image; bottom row: ELA map

3) Use in AI detection: The use of ELA for AI detection is motivated by the observation that generated content does not undergo the same compression as real digital images do. Generators often output uncompressed images or images in formats that use lossless compression, which lack the irregular error traces introduced by real-world image compression.
To perform classification, ELA is calculated and saved for each image in the dataset prior to the training process. Though ELA images are sometimes enhanced in contrast for visualization, the raw output of the extraction process is fed to the network for training. The same custom CNN as in figure 2 learns to distinguish the subtle differences in error maps, which are normally very difficult to spot with the naked eye. The resulting model is a binary classifier, outputting predictions on whether a given image is real or not.

C. Latent Reconstruction Error Based Method for generated Image Detection (LaRE^2) [6]

1) Theoretical background: In recent years, the latent space of deep generative models has emerged as a powerful domain for image analysis, including classification, manipulation, and detection tasks. The concept of latent space refers to an abstract representation of data, often reduced in dimensionality, in which essential features are retained and irrelevant detail is discarded. In the context of AI image generation and detection, latent spaces offer a size-efficient way of manipulating and storing data, which has proven to have a positive effect on both computational complexity and training accuracy.
The hypothesis that is the basis for the creation of reconstruction error-based detection is the idea that AI-generated images, being part of the generator's distribution, are easier to reconstruct than real images. As such, the reconstruction error should be significantly smaller for synthetic images than for real images. First proposed specifically for detecting images created by diffusion models in DIRE [11], and later refined by Luo in LaRE^2 [6] by reducing the process complexity and computing the errors in latent space. In the DIRE process, both real and fake images are taken through a latent-space inversion and reconstruction pipeline - this means the error is calculated in pixel space by taking the difference between the original and reconstructed image. In the improved version, the computation is done in the latent space entirely, improving the process efficiency and reducing complexity.
This work adopts the latter approach through the use of the LaRE^2 framework.

2) Overview of the method: LaRE^2 (Latent Reconstruction Error + Error-Guided Feature Refinement) is a state-of-the-art method developed for the purpose of diffusion-model-generated image detection. It avoids entirely reconstructing the image and instead computes the denoising prediction error in latent space, using a single-step reverse diffusion operation. The proposed method is comprised of two components:

- **Latent Reconstruction Error (LaRE)** is the first stage of the process, used solely to compute the latent space representation of the reconstruction error map for the dataset images.
- **Error-guided Feature refinement (EGRE)** comprises two modules ESR and ECR, and enhances the detection process. The error-guided spatial refinement module (ESR) spatially aligns LaRE to the intermediate feature map of the CNN, allowing it to emphasize regions that convey more information to the detector. Error-Guided Channel Refinement module (ECR), which learns to emphasize feature dimensions that correlate with high and low reconstruction error.

IV. Dataset

The dataset is composed of five subsets, each corresponding to one of the generators included. Each subset consists of 5000 images divided evenly into 2500 real images and 2500 fakes. The real images all come from the same source, while the generated images were generated by the authors of the GenImage [12] dataset using a corresponding model, with the exception of FLUX, which was generated by us. The total, therefore, is 12,500 real images and 12,500 synthetic ones.
These subsets were further divided into the training and validation parts. Each of these contains an even split of real and fake data, with the distinction of the training set having four-fifths of the data and validation having the rest.

A. Included Generative models

This dataset includes five distinct generative models, each of which boasts a unique architecture, which allows us to measure the impact of each generative approach on the investigated detection methods. Figure 4 shows examples of images from the dataset.

- **ADM (Ablated Diffusion Model) [8]** is a diffusion model that, unlike many contemporary models, operates directly on pixel space. It is widely considered a foundational benchmark for diffusion models, which makes it a good candidate for assessing detection methods.
- **BigGAN [7]** is a large-scale Generative Adversarial Network, which offers architectural diversity in this dataset, letting us see how well the detection methods work on a model with a completely different architecture from the rest.
- **FLUX [10]** is a text-to-image model developed by Black Forrest Labs, known for its high output quality and fidelity. It uses a state-of-the-art hybrid architecture, combining transformer-based techniques with diffusion models.
- **Midjourney [9]** is a proprietary commercial generative model known for its distinct visual style. While the architecture details are not disclosed, it is believed to utilize a diffusion approach, similar to Stable Diffusion. Its prevalence in creative industries makes it a valuable benchmark for discriminators, as this model is likely to be encountered outside of the testing environment.
- **Stable Diffusion v1.5 [2]** is an open-source, latent space diffusion model known for efficient generation of high-quality images. Currently it is one of the most easily accessible and widespread models available.

B. Generated Image Sources

The four models taken from GenImage (ADM, BigGAN, Midjourney, and Stable Diffusion v1.5) were all created by the authors in the same way to ensure consistency. A total of 1000 class labels were taken from ImageNet [15], a large-scale dataset containing real-world photographs on a variety of topics. For each of the classes, every generator was prompted to create over 160 images. "Photo of class"was the model input sentence in each instance, with the selected label used in place of 'class'.

Data for the synthetic part of the FLUX subset was generated locally, using the same prompting style and class labels as the original GenImage dataset.

C. Real Image Sources

All images in this dataset were taken from GenImage, which in turn were curated from ImageNet. Just like the generated portion, each of the selected real images corresponds to one of the 1000 labels. A further subset of the GenImage dataset called 'tiny GenImage' was selected to better suit our computational resources. The creators of tiny GenImage took care in selecting a balanced amount of images from each class so as not to introduce any bias. Despite the fact that true images come from the same source for each subset, each real image is unique across the entire dataset, which enables cross-model and multi-model training without the risk of overfitting or data imbalance.

V. Experimental setup

The primary objective of the experiments presented in this work is evaluating the effectiveness of three approaches:

Rysunek 4. Examples of images from the dataset

PRNU-based, ELA-based and LaRE-based—to distinguishing between real camera-taken images and AI-generated images The experiments were designed to measure classification performance of each method, gauge generalization for previously unseen models, and compare robustness under diverse training and testing scenarios.

The evaluation aims to investigate these specific areas:

- **Single model performance** - The ability to detect images created on the same model as the detector has been trained on.
- **Cross-model performance** - How well a detector trained on one selected generator (e.g., BigGAN-only) performs when tasked with classification on the other models, unseen during training. This reflects real-world conditions, where the source of the image is likely unknown.
- **Multi-generator performance** - How robust is a method when trained on a combined dataset of all five methods. This combination would be used to create a general-purpose detection system.
- **Method strengths and tradeoffs** - Identification of strengths, weaknesses and scenarios in which each method tends to underperform. Includes observations about types of generators that a method may struggle with, method complexity, and time tradeoffs.

By evaluating each method in consistent conditions, using seeded values and repeatable tests, we aim to provide a fair comparison of method performance using different sources of generated images and varied evaluation scenarios.

A. Training Setup

Each of the three methods was trained using the following two configurations:

- **Per-generator training** consists of five training runs, during which the detector is trained on one of the five models from the dataset.
- **Combined training** featured only one training run, in which the training and validation data had been concatenated and shuffled. This setup allowed the trained model to learn more general features of the generator models, possibly leading to better overall detection accuracy.

The result of the training is 18 trained models, 6 (5 per-generator, 1 combined) for each detection method.

Tabela I
CNN TRAINING HYPERPARAMETERS FOR PRNU AND ELA

Hyperparameter	**Value**
Input size	$128 \times 128 \times 3$ px
Conv2D filters	$[32, 64, 64]$
Conv2D Kernel size	3×3
Activation functions	ReLU (hidden layers), Sigmoid (output)
Pooling	MaxPooling 2×2
Dense layers	$[64, 1]$
Loss function	Binary cross-entropy
Learning rate	1×10^{-4} (default)
Optimizer	Adam (β_1=0.9, β_2=0.999, $\epsilon = 1 \times 10^{-7}$)
Batch size	32 (default)
Epochs	30
Validation split	20% (via `train_test_split`)
Evaluation metrics	Accuracy, Precision, Recall, F1, ROC AUC, AP
Checkpoint monitor	Validation Accuracy

Tabela II
TRAINING HYPERPARAMETERS FOR LARE

Hyperparameter	**Value**
Model architecture	`CLipClassifierWMapV6`
CLIP visual backbone	`RN50`
Input resolution	224×224 px
Epochs	50
Batch size (train)	48
Batch size (test)	40
Initial learning rate	1×10^{-4}
Optimizer	Adam (β_1=0.9, β_2=0.999, $\epsilon = 1 \times 10^{-7}$)
LR scheduler	ReduceLROnPlateau (factor 0.5, patience 4)
Gradient clipping	$g_2 \leq 5.0$
Loss function	Cross-Entropy (no label smoothing)
Data augmentation	Pad $\rightarrow$ RandomCrop (strong aug. disabled)
Validation method	Contrastive (`con`)
Validation ratio	0 (separate val file)
Dataloader workers	8
GPU configuration	Single GPU (ID 0)

B. *Evaluation Setup*

The models obtained from training have been evaluated in three categories corresponding to the areas outlined in the beginning of this section: single-model tests focusing on data from one model, cross-model tests evaluating performance of using data from one model on others, and multi-generator tests combining data from all available models for the most comprehensive detector. In total, 108 test cases have been conducted.

VI. RESULTS AND ANALYSIS

The results from each test were evaluated using industry-standard metrics such as accuracy, average precision (AP), area under ROC curve (AUC), recall, F1 score, and loss [14] [13]. All results were presented in figures 5, 6, 7. Analysis of the performance is presented in the following sections.

A. *Overview of Results*

1) Single-model performance: The results of tests on models trained and tested on the same generative model are overwhelmingly positive. Nearly 100% accuracy, AP and AUC for each method and model combination indicate that this is an area that does not need any significant improvements or breakthroughs. The worst outcomes in this category come from a PRNU-based detector trained on Midjourney, with 90% accuracy and a 90% F1 score. While still impressive, these results are almost 10% worse than other cases, which indicates that some part of the proprietary Midjourney architecture might interfere with PRNU extraction. Results obtained by recent studies in the area seem to follow the same pattern of excellent single-model performance, meaning that a model that underperforms in this category most likely has serious issues with implementation or method.

2) Cross-model performance: Cross-model generalization is an area where the models struggle the most. The results vary wildly depending on the method, training dataset and testing dataset. The impact of generator architecture is most evident on models trained on BigGAN data, which are almost entirely unable to properly classify any other model than itself, regardless of the method used. Interestingly, while PRNU-based and ELA-based models struggle with detecting BigGAN when trained on other generators, LaRE-based detectors seem to correctly classify the unseen GAN to a surprisingly high degree.

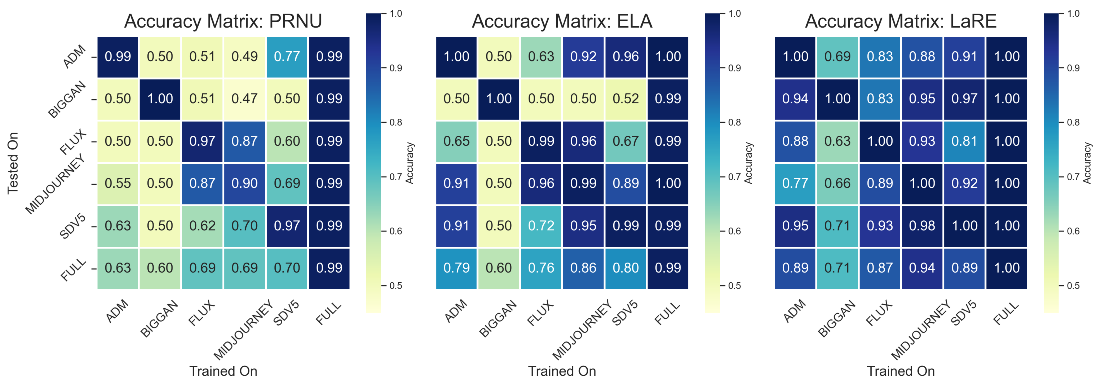
It also appears that models that have similar architectures have an easier time detecting each other than other models. FLUX, Midjourney and Stable Diffusion share the most similarities in their architecture, and as such, they have consistently the highest performance in the cross-model tests. Notably, in LaRE-based tests, whether one of the three mentioned models is being tested or trained on has little impact on the results, unlike BigGAN, for which, as previously mentioned, the trained-on and tested-on results differ greatly.

Overall, the best results in this category came from LaRE-based detectors, which provided satisfying performance in most cases. ELA had slightly worse results than LaRE, but when we consider that data preparation and training are several times faster than LaRE, it might open up some niche scenarios where the tradeoff for speed is worth the slight decrease in performance. PRNU performed the worst of the three methods, which could be a reason to forgo further research on PRNU-based detection for better-performing alternatives.

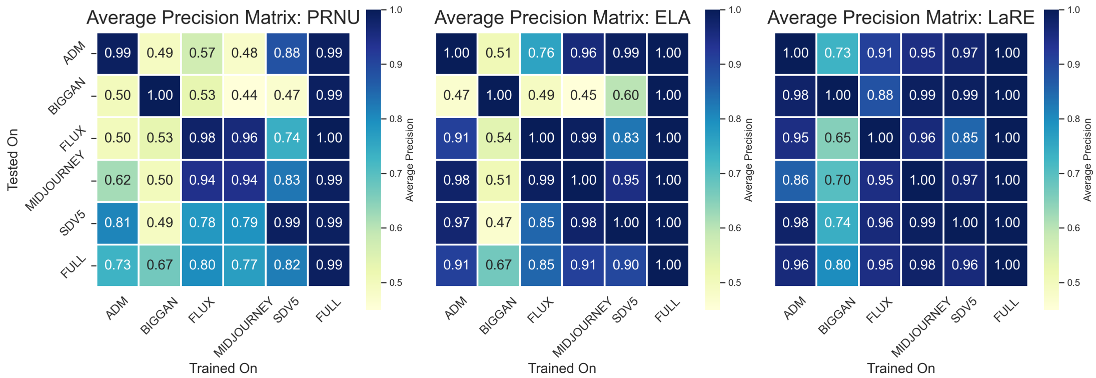
3) Multi-model performance: While the results of the previous section were somewhat worrying, the performance of the models trained on all available generators surpassed even single-model tests. In all tested scenarios, the models tested on full data have showcased nearly perfect detection, with accuracy never dropping below 99%. There are no notable outliers in this section, but it is worth pointing out that despite underperforming in the previous section, PRNU matches the effectiveness of ELA and LaRE in this case. The multi-model dataset is marked as FULL in the figures 5, 6 and 7.

B. *Discussion*

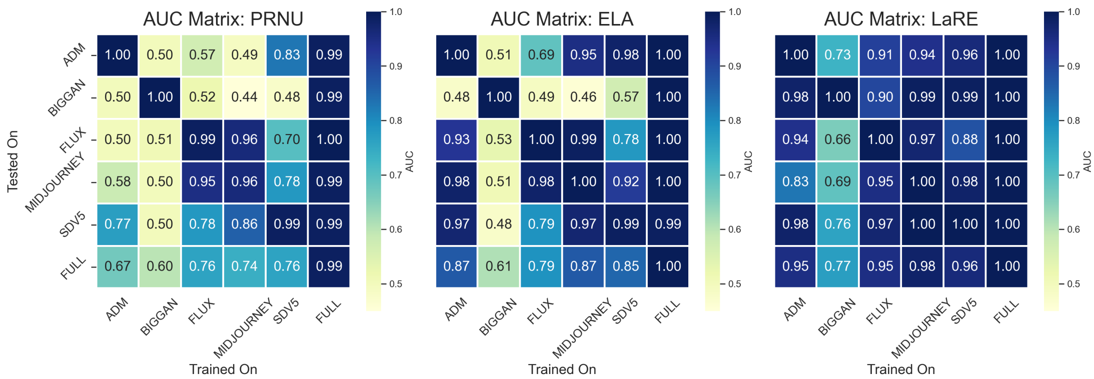
As previously mentioned, the models have performed the best in the single-model and multi-model cases. In both cases, however, they have seen the models they were going to detect during training. While during testing it might seem the models trained on data from all generative models might be sufficient detectors for real-world scenarios, we do not



Rysunek 5. Accuracy on training validation for all three methods



Rysunek 6. Average Precision on training validation for all three methods



Rysunek 7. AUC on training validation for all three methods

know how these models would react to previously unseen data. Based on the cross-model tests, we can reasonably assume that even the most comprehensive models from this paper might struggle against data from an unseen model, with the degree of accuracy heavily dependent on the architecture of the unseen model. It is therefore better to focus on the results of cross-model tests, since, in a real-world scenario, it is infeasible to create and maintain a dataset containing all the models a detector can encounter.

Notably, the performance of the LaRE-based detectors on the BigGAN dataset was a lot better than expected. It was theorized that due to the differences in GAN architecture from diffusion models, for which the method was developed, LaRE-based detectors might perform worse than ELA- and PRNU-based detectors. It appears, however, that despite the differences, the LaRE^2 framework still outperformed the other two methods on the BigGAN dataset.

VII. Conclusions

The results obtained from the tests confirmed that models trained and tested on the same model perform near-perfect detection, as proposed in previous related works. The cross-model results revealed that models with similar architectures (e.g., Midjourney and Stable Diffusion) have an easier time detecting each other than less related models (e.g., FLUX and BigGAN). In this category, LaRE-based detectors performed the best across all metrics and models, while PRNU-based detectors performed the worst. Despite some subpar results in the second category, every single method exhibited overwhelmingly positive performance when trained on the subset containing data from all generative models.

Based on the obtained results, there are several areas worth investigating further. The introduction of more varied generative models into the dataset could shed more light on the exact characteristics of models that cause problems in cross-model generalization. Increasing the number of images per generator could also dispel doubts about overfitting the training data, which is possible for relatively small datasets.

As for the future work, to further improve non neural network based extraction methods, the simple pipeline for the PRNU- and ELA-based models could be expanded further by the use of more comprehensive preprocessing techniques. The conducted tests can be further extended by using bigger datasets or by introduction the most common modifications present in the everyday photography.

LITERATURA

[1] Goodfellow, I.J. et al.: Generative adversarial networks. arXiv preprint arXiv:1406.2661 (2014)

[2] Rombach, R. et al.: High-resolution image synthesis with latent diffusion models. In: CVPR, pp. 10684–10695 (2022)

[3] Martin-Rodriguez, F., Garcia-Mojon, R., Fernandez-Barciela, M.: Detection of AI-created images using CNNs. Sensors **23**(22), 9037 (2023). 10.3390/s23229037

[4] Krawetz, N.: A picture's worth... In: Black Hat DC (2008), https://blackhat.com/presentations/bh-dc-08/Krawetz/Whitepaper/bh-dc-08-krawetz-WP.pdf, last accessed 2025/05/29

[5] Lukas, J., Fridrich, J., Goljan, M.: Digital camera identification from sensor pattern noise. IEEE Trans. Inf. Forensics Secur. **1**(2), 205–214 (2006). 10.1109/TIFS.2006.873602

[6] Luo, Y. et al.: LaRE^2: Latent reconstruction error for diffusion-generated image detection. arXiv preprint arXiv:2403.17465 (2025)

[7] Brock, A., Donahue, J., Simonyan, K.: Large scale GAN training. arXiv preprint arXiv:1809.11096 (2019)

[8] Ho, J., Salimans, T.: Classifier-free diffusion guidance. In: NeurIPS Workshop on Score-Based Methods (2022)

[9] Midjourney: Midjourney image generation platform. https://www.midjourney.com, last accessed 2025/05/29

[10] Black Forest Labs: FLUX: High-fidelity image generation. https://bfl.ai/, last accessed 2025/05/17

[11] Wang, Z. et al.: DIRE for diffusion-generated image detection. In: ICCV, pp. 22445–22455 (2023)

[12] Zhu, M. et al.: GenImage: A million-scale benchmark for detecting AI-generated image. arXiv preprint arXiv:2306.08571 (2023)

[13] Sokolova, M., Lapalme, G.: Performance measures in classification. Inf. Process. Manag. **45**(4), 427–437 (2009). 10.1016/j.ipm.2009.03.002

[14] Fawcett, T.: An introduction to ROC analysis. Pattern Recogn. Lett. **27**(8), 861–874 (2006). 10.1016/j.patrec.2005.10.010

[15] Deng, J. et al.: ImageNet: A large-scale hierarchical image database. In: CVPR, pp. 248–255. IEEE (2009). 10.1109/CVPR.2009.5206848

[16] Yan, S., Li, O., Cai, J., Hao, Y., Jiang, X., Hu, Y., Xie, W.: A Sanity Check for AI-generated Image Detection. arXiv preprint arXiv:2406.19435 (2025). https://arxiv.org/abs/2406.19435

[17] Abbasi, M., Váz, P., Silva, J., Martins, P.: Comprehensive Evaluation of Deepfake Detection Models: Accuracy, Generalization, and Resilience to Adversarial Attacks. Applied Sciences 15(3), 1225 (2025). 10.3390/app15031225

[18] Li, B., Sun, J., Poskitt, C.M., Wang, X.: How Generalizable are Deepfake Image Detectors? An Empirical Study. arXiv preprint arXiv:2308.04177 (2024). https://arxiv.org/abs/2308.04177

[19] Yang, S., Guo, H., Hu, S., Zhu, B., Fu, Y., Lyu, S., Wu, X., Wang, X.: CrossDF: Improving Cross-Domain Deepfake Detection with Deep Information Decomposition. arXiv preprint arXiv:2310.00359 (2024). https://arxiv.org/abs/2310.00359

[20] Wu, S., Liu, J., Li, J., Wang, Y.: Few-Shot Learner Generalizes Across AI-Generated Image Detection. arXiv preprint arXiv:2501.08763 (2025). https://arxiv.org/abs/2501.08763

[21] Roca T., Cintron Roman A., Torres Vega J., Duarte M., Wang P., White K., Misra A., Lavista Ferres J.:How good are humans at detecting AI-generated images? Learnings from an experiment arXiv preprint arXiv:2507.18640 (2025). https://arxiv.org/abs/2507.18640

[22] Konstantinidou D., Karageorgiou D., Koutlis C., Papadopoulou O., Schinas E., Papadopoulos S.:Navigating the Challenges of AI-Generated Image Detection in the Wild: What Truly Matters? arXiv preprint arXiv:2507.10236 (2025). https://arxiv.org/abs/2507.10236