AI

Article

Detection of AI-Generated Synthetic Images with a Lightweight CNN

Adrian Lokner Lađević [1], Tin Kramberger [1], Renata Kramberger [1] and Dino Vlahek [2,*]

[1] Department of Information Technology and Computing, Zagreb University of Applied Sciences, Vrbik 8, 10000 Zagreb, Croatia; adrian.lokner.ladjevic@gmail.com (A.L.L.); tin.kramberger@tvz.hr (T.K.); renata.kramberger@tvz.hr (R.K.)

[2] Faculty of Electrical Engineering and Computer Science, University of Maribor, Koroška Cesta 46, 2000 Maribor, Slovenia

* Correspondence: dino.vlahek1@um.si

Abstract: The rapid development of generative adversarial networks has significantly advanced the generation of synthetic images, presenting valuable opportunities and ethical dilemmas in their potential misuse across various industries. The necessity to distinguish real from AI-generated content is becoming increasingly critical to preserve the integrity of online data. While traditional methods for detecting fake images resulting from image tampering rely on hand-crafted features, the sophistication of manipulated images produced by generative adversarial networks requires more advanced detection approaches. The lightweight approach proposed here is based on convolutional neural networks that comprise only eight convolutional and two hidden layers that effectively differentiate AI-generated images from real ones. The proposed approach was assessed using two benchmark datasets and custom-generated data from Sentinel-2 imagery. It demonstrated superior performance compared to four state-of-the-art methods on the CIFAKE dataset, achieving the highest accuracy of 97.32%, on par with the highest-performing state-of-the-art method. Explainable AI is utilized to enhance our comprehension of the complex processes involved in synthetic image recognition. We have shown that, unlike authentic images, where activations often center around the main object, in synthetic images, activations cluster around the edges of objects, in the background, or in areas with complex textures.

Keywords: convolutional neural networks; generative adversarial networks; classification; synthetic images; explainable artificial intelligence

check for updates

Citation: Lokner Lađević, A.; Kramberger, T.; Kramberger, R.; Vlahek, D. Detection of AI-Generated Synthetic Images with a Lightweight CNN. *AI* **2024**, *5*, 1575–1593. https://doi.org/10.3390/ai5030076

Academic Editor: Demos T. Tsahalis

Received: 8 August 2024
Revised: 27 August 2024
Accepted: 30 August 2024
Published: 3 September 2024

1. Introduction

The remarkable progress in deep learning, specifically in Generative Adversarial Networks (GAN) [1], in recent times has enabled the generation of synthetic images of such high quality that they can often be mistaken for genuine images [2]. While this advancement opens up exciting opportunities across various industries, it simultaneously raises significant ethical issues due to the possibility of misuse. Such misapplications could have profound societal impacts, including disseminating false information, engaging in fraud, committing identity theft, and generating harmful or inappropriate content [3]. As deep learning evolves at an unparalleled pace and AI-generated images become more lifelike by the day, establishing robust techniques to differentiate between real and AI-created content becomes imperative. This is essential to counteract the adverse effects and maintain the trustworthiness and integrity of online information.

Other than visual inspection, an unreliable method for determining fake images today, there are two main approaches for detecting fake images. These include hand-crafted feature extraction using image processing techniques and deep learning [2]. Early methods for detecting fake images resulting from image tampering aimed to identify a specific tampering technique, such as splicing or copy–move [4]. These tampering techniques typically involve altering certain parts of an image to create or modify objects within it. Early fake-detection approaches were usually based on extracting features from the frequency domain. For example, the approach proposed in [5] divides the image into overlapping blocks and detects the copy–move forgery by matching the features extracted from these blocks using discrete cosine transformation. Similarly, the method proposed in [6] for feature extraction from the frequency domain obtains low-frequency components using the discrete wavelet transformation. At the same time, singular value decomposition is applied to these components to obtain the feature vectors. However, this method is time-consuming and sensitive to image objects that are rotated, scaled, or blurred. To improve the efficiency of the previous method, the same author proposed in [7] the usage of Fourier–Mellin transformation. Transformation is used to mitigate sensitivity to geometric operation while the whole detection process is accelerated using the Bloom filter. Nevertheless, when introduced to image splicing, the mentioned methods perform poorly, as image splicing involves integrating segments from different sources, each with distinct textures and features. The method proposed in [8] extracts the color filter array pattern from the image. The statistical analysis of local inconsistencies within these patterns is then performed to differentiate between original and fake regions. In [9], the classification model is used to identify irregularities introduced by splicing in images, thereby identifying the fake ones. Feature extraction is performed by a discrete cosine transformation, as splicing often disrupts the frequency patterns of the original image, while Markov features are derived from obtained transform coefficients.

However, with the development of sophisticated GANs, images are often altered using multiple tampering techniques simultaneously, creating more realistic-looking images that are difficult to identify as manipulated, making it challenging to pinpoint both the nature of the tampering and the specific regions affected in the image. Thus, the previously effective methods based purely on feature extraction can no longer accurately detect these manipulations [10]. This can be overcome with deep learning approaches such as convolutional neural networks (CNNs). Convolutional neural networks are centered around various architectures of deep artificial neural networks featuring convolutional and pooling layers followed by several hidden layers of neurons. These layers progressively extract higher-level features from the raw input, such as images. By enhancing the quantity of the layers, these methods can approximate more intricate decision functions, thereby attaining superior classification accuracy [11,12]. Nevertheless, today, high-performing CNNs, such as VGGNet [13], DenseNet [14], and ResNet [15], consist of a high number of layers that increase the need for a large quantity of data and the complexity of the training procedure due to the presence of multiple local optima and an extensive number of hyperparameters. Additionally, they are also considered to be black-box function approximates that do not allow for an explanation of their decisions [16]. For example, VGGNet consists of 16 layers, including 13 convolutional layers and 3 fully connected layers with several thousands of neurons, while ResNet and DenseNet are more than 100 layers deep. In contrast, a lightweight CNN for image forgery detection was proposed in [17], which includes a standard 3×3 convolutional layer followed by 17 bottleneck layers of 1×1 and 3×3 convolutions. Compared to a standard convolution operating across a large kernel, using small kernels such as 1×1 enables higher computational efficiency. Another lightweight CNN with 3 convolutional layers was proposed in [18], with 32, 64, and 128 filters, respectively, and a 3×3 kernel size for forgery detection in images. However, current lightweight methods cannot extract high-quality features from input images compared to deep CNNs due to a small number of layers and kernel sizes, thus not allowing for high classification performance.

Central to our methodology is a deep learning approach based on CNN architecture. Several important scientific contributions with multidisciplinary and social implications arise from this study. These contributions include the following:

- A high-performance CNN-based method consisting of only eight convolutional and two hidden layers enhances the human ability to identify AI-generated images using

computer vision. The method was evaluated on two benchmark datasets as well as on custom-generated data based on Sentinel-2 imagery;
- An approach for creating high-quality synthetic visual content through transfer learning based on publicly available StyleGAN framework [19]. Specifically, we generated synthetic satellite images based on Sentinel-2 imagery for this study;
- The utilization of Explainable AI (XAI) to deepen our understanding of the intricate processes involved in synthetic image recognition.

The mentioned scientific contributions represent a significant leap in addressing the intricate challenges arising from the rapid development of modern technology. This progress holds vital implications for upholding the trustworthiness and accuracy of data, which is crucial in an era heavily reliant on digital information.

The rest of the paper is structured as follows. Section 2 provides background on synthetic image generation and detection algorithms. In the most research-intensive Section 3, we first summarize creating synthetic data using StyleGAN. In continuation, we described the proposed CNN architecture. In Section 4, we show the results and discuss the work carried out, as well as the interpretation of the resulting model.

2. Related Work

2.1. Synthetic Image Generation

The core idea of GANs encompasses an architecture of two distinct neural networks: a generator and a discriminator [1]. The role of the generator is to produce new data instances, such as images, using the provided input data. Meanwhile, the discriminator's task is to assess these instances to determine whether they are authentic or fabricated by the generator. Essentially, this creates a competition where the generator aims to generate data that are indistinguishable from actual data to deceive the discriminator. In contrast, the discriminator's objective is to accurately identify which data are authentic and which have been created by the generator. Both real and synthetic images are fed to the discriminator during training, after which a loss value is computed.

One of the most popular GAN architectures is StyleGAN, introduced in [19]. It begins by processing the input features by the mapping network comprised of eight fully connected layers, followed by an affine transformation. This mapping aims to create features that allow the generator to perform efficient rendering and avoid feature combinations that do not occur in the training dataset. At the same time, affine transformation enables the control of image generation by scaling, rotating, translating, mirroring, and shearing image objects while preserving their original relationships. The random noise is then added to the output of this mapping, which is then additionally processed by adaptive instance normalization (AdaIN) and the convolutional layers. In this step, image generation starts with a low-resolution image and gradually adds more layers to increase the resolution and add details (e.g., from 4×4 to 1024×1024). Two issues with the images created by the first version of StyleGAN are blood-like artifacts and low shift-invariance that manifest in an unnatural image alignment of some objects in the image. This is overcome with the improved architecture of the StyleGANv2, proposed in [20]. The authors observed that the blob artifacts were associated with the AdaIN normalization, so they introduced a weight-demodulation process instead that is directly applied to the weights of the convolution layers. On the other hand, the issue of shift-invariance arose from progressive growth. StyleGANv2 removes the progressive growth by learning directly at the final resolution from the beginning. Another improvement of StyleGAN architecture comes with StyleGANv3 [21], which introduces Fourier features and adds an affine transformation layer to these features for more efficient translation and rotation while adding additional minor changes to the architecture, such as new upsampling and downsampling filters, to increase generated image quality. Nonetheless, the high quality of StyleGANv3 comes at a price. It demands more time, greater computational resources, and a larger dataset to achieve satisfactory results, as it struggles with mode collapse. This phenomenon causes the generator to produce a limited variety of images, resulting in reduced diversity in the output. In [22], the authors proposed a module that can be added to various GAN architectures that mitigates the mode collapse by penalizing the generator when it generates images with similar features. Another strategy that addresses the mode collapse issue is using multiple generators [23] or modifying the discriminator's behavior [24].

Numerous alternatives to StyleGAN architectures for synthetic image generation are available today. These alternatives offer different capabilities and advantages, depending on specific applications and requirements. One example is the deep convolutional generative adversarial network proposed in [25]. This network comprises only convolutional layers without max-pooling or fully connected layers, leading to more stable training. However, training requires a large amount of high-quality data. In the case of limited data, it suffers from mode collapse. Additionally, stable diffusion methods are designed to address these issues of mode collapse [26]. These methods use an encoder–decoder structure, where the encoder captures essential features while reducing dimensionality, and the decoder reconstructs the image. On the reduced feature space, Gaussian noise is iteratively applied for the diversity of generated images. The encoder consists of several convolutional layers that progressively downsample the image, capturing different levels of abstraction. Simultaneously, the decoder mirrors the encoder but in reverse, focusing on generating detailed and realistic images. Nevertheless, all techniques add additional computational burden to the training process, increasing the training time.

The CIFAKE dataset used in this research was created using diffusion models, which utilize an encoder–decoder structure in an ANN. In simple terms, diffusion models compress images into a lower-dimensional feature space, add noise to produce new content, and then map them back to the high-resolution space. On the other hand, generating a synthetic image dataset using GAN involves adversarial training, where images are created directly from a high-dimensional feature space that is extracted from input images. The adversarial process includes generating images and then refining them based on feedback from a discriminator. Both GAN and diffusion-based image generation produce highly realistic and high-quality synthetic images.

2.2. Synthetic Image Detection with CNN

As CNNs have automatic feature extraction capabilities through convolutional operations, they are widely used in image analysis and object detection today. Initially, CNNs were developed for tasks such as handwritten zip code recognition [27] and handwritten digit recognition [28], eliminating the need for manual feature extraction. However, the computational limitations of 1990s hardware restricted their effectiveness. The advancement of hardware in the 2010s and the availability of powerful graphics processing units (GPUs) designed for parallel processing revitalized CNN research [29]. In [30], the authors investigated the ability of a CNN comprised of eight layers (including an input layer, two convolutional layers, two max-pooling layers, and three fully-connected layers) to extract features for fake image detection automatically. While this method has demonstrated high accuracy in detecting manipulated images, it has only been tested on one dataset, so its robustness still needs to be verified. In contrast, VGGNet [13] is recognized for its reliable performance across different benchmarks and datasets [31]. Its simple architecture consists of 13 convolutional layers and 3 fully connected layers, making it a reliable option for feature extraction and transfer learning. However, it demands substantial computational resources for training and utilizes only small 3 $\times$ 3 convolutional filters, which may not be as efficient in capturing high-level features as newer architectures. Similarly, the network described in [15] utilizes 3 $\times$ 3 convolutional filters while implementing residual learning via residual layers. Each residual layer contains one or more convolutions and includes a shortcut (or skip) connection that bypasses one or more layers. It can be extremely deep, consisting of over 100 layers, effectively managing them through their residual connections. On the other hand, in [14], a densely connected convolutional network is introduced. This network is built on the concept of the dense connectivity of layers, as opposed to bypassing them. Dense connectivity means that each layer is connected to every other layer in a feed-forward fashion. This leads to improved interpretability, as each layer has direct access to the loss function gradients and the enhanced efficiency and performance of deep networks. Despite achieving high classification accuracies, both methods have drawbacks, such as high computational and memory requirements and susceptibility to overfitting. The approach presented in [32] encounters challenges similar to densely connected convolutional networks, such as issues with computational efficiency and potential overfitting. In this neural network architecture, both images and their Fourier frequency decomposition are used as inputs. The method involves combining the spatial representations of images with their frequency domain representations before passing them through fully connected layers. Integrating multiple types of data representations adds complexity to the model and its training process. Additionally, this approach requires fine-tuning a large number of hidden parameters to achieve optimal results.

2.3. Explainable AI

The understanding of decision-making principles behind early machine learning approaches is straightforward. For instance, decision trees can be analyzed by traversing their nodes [33], while linear models can be interpreted by examining their parameters [34]. However, due to their simplicity, these models achieve lower classification accuracies than neural network-based [35] models, which are usually considered black-box approximators due to their non-linearities [16,36]. Nevertheless, recent attempts have been made towards interpreting black-box approaches, such as learning interpretable models locally around a given sample [37] or estimating each feature's importance on the output classification [38]. Despite being applicable across linear, non-linear, and deep models (such as CNNs), these techniques are sensitive to high correlation among features, and they do not consider the issue of learned features that do not support the interpretation of the dependencies between them [39].

One approach to overcome this challenge is visualizing the decision-making process for CNNs, often by generating heatmaps from input samples to highlight the observed neurons' activations [40–42]. In [40], the gradient of the loss function is calculated for the class of interest with respect to the pixels in the input image. Then, these gradients are visualized as a heatmap representing the input features, highlighting either the absolute values or contrasting negative and positive contributions. Similarly, the technique proposed in [41] is used to visualize feature activations by mapping activations back to the input space. The idea is to reverse the operations of a CNN, such as pooling and convolution, to trace back from the output to the input space. Unlike the previous two methods, in [42], the gradient is not backpropagated to the image but to the last convolutional layer. These gradients provide valuable insight into how changes in the feature maps directly influence the class score, as the weighted feature map is calculated by evaluating the positive impact of individual pixels within the feature maps on the overall class score. The derived weights are then used to compute a weighted summary of the feature maps, producing a class-specific heatmap. This heatmap effectively showcases the critical areas within the input image that significantly contribute to the prediction of the class.

3. Materials and Methods

In this section, we present a CNN-based method that consists of eight convolutional and two hidden layers. We begin with data preparation, followed by a detailed explanation of how to create synthetic datasets for fake image detection containing both generated and real satellite images using GAN architecture. Additionally, we discuss using GAN architecture along with training details and a quantitative evaluation of generated image quality using standard metrics.

3.1. Data Preparation

For this study, we have created a new dataset specially optimized for GAN training. Firstly, we collected images portraying various locations from across the globe, categorized into cities, coastal areas, deserts, lowlands, and tundras from the Copernicus Data Space Ecosystem [43]. Figure 1 displays sample satellite images for each category. This diversity ensures that the GAN learns to generate a wide range of synthetic satellite images with different features of terrain, vegetation, or human settlements.

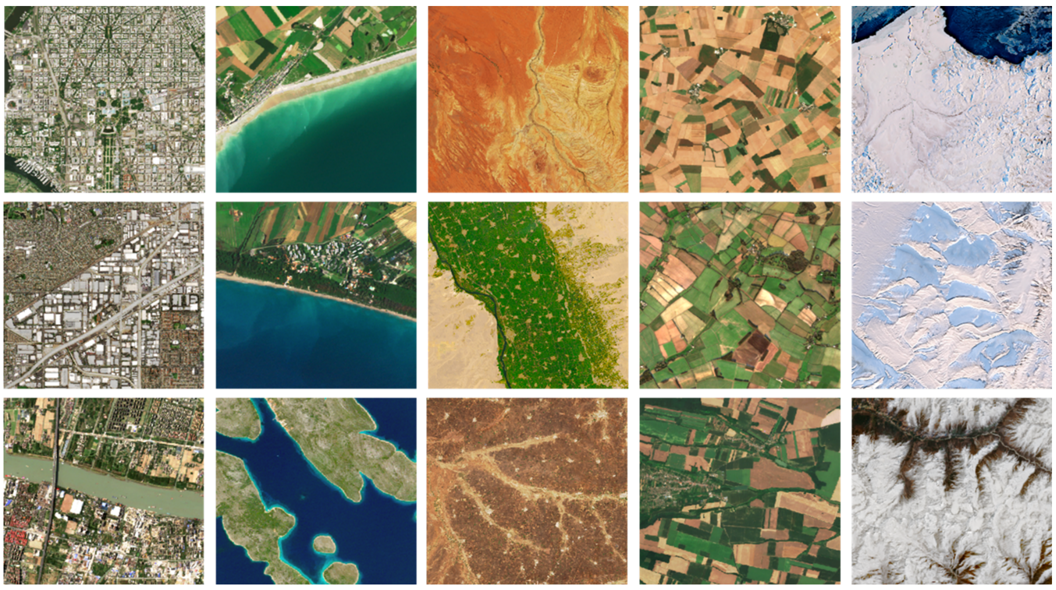


Figure 1. Example of collected satellite images.

We extracted slices of size 256 × 256 pixels from a satellite image in GeoTIFF format. Each of these slices represents a separate and unique geographic entity within a larger satellite image, allowing for an increasing variety and amount of data for training a GAN by generating extra images from a single original satellite image (see Figure 2).

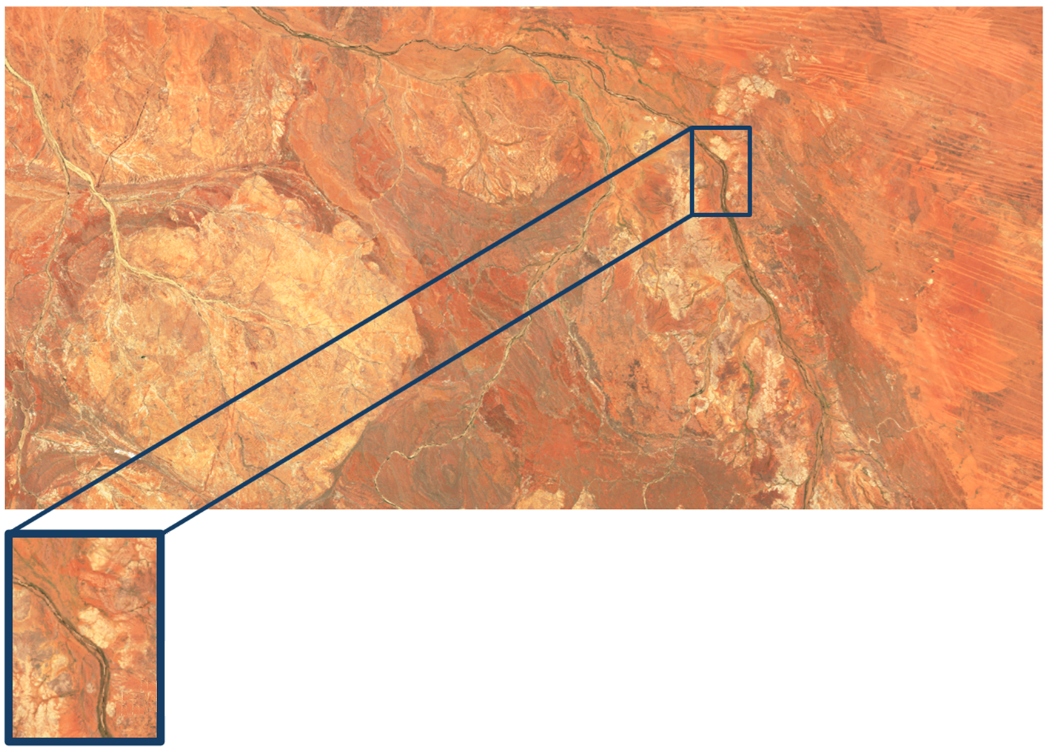


Figure 2. Example of slice extraction.

After acquiring the slices, it is crucial to carefully review and select the proper ones to be used for GAN training. As satellite images often contain disturbances such as distortion, unwanted objects, clouds, inadequate lighting, and other factors that impair image quality, including such images can cause the model to learn to reproduce these disturbances instead of focusing on the key features and details that need to be generated. Finally, after the cleaning, we acquired 50,000 slices of satellite images in PNG format. Additionally, to increase the dataset's scope, 90° rotations and vertical and horizontal mirroring were applied to the collected images. Several examples of such transformations are shown in Figure 3.

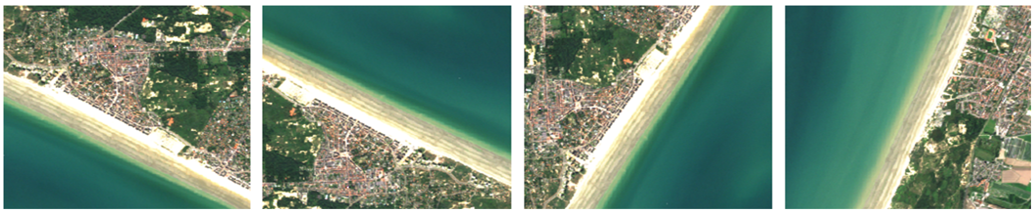


Figure 3. Examples of image transformations.

Overall, the GAN training dataset consisted of 100,000 images, of which 50,000 were collected, and 50,000 were created by transformations of the collected images, thus ensuring a sufficient amount of diverse data for efficient learning and generating new satellite images.

3.2. Synthetic Data Generation

In order to generate synthetic satellite images, we rely on StyleGANv3 architecture, which is an advanced version of the StyleGAN framework created and released by NVIDIA [21]. StyleGANv3 is recognized for its remarkable capacity to produce top-notch images. The continuation of this subsection presents the step-by-step procedure for setting up and training a GAN model using images collected and processed, as presented in the previous subsection.

The training of a GAN can be demanding and time-consuming, as it depends on the complexity and size of the dataset and the available computing resources. During training, unwanted features can appear in the generated images, usually due to insufficient data quality, poor neural network architecture, or inappropriate hyperparameters such as learning rate, batch size, and the number of training epochs. One of the widespread challenges during the training of GANs is mode collapse. Mode collapse is a problem wherein the generator begins to generate images with limited variety, focusing only on certain features of the training data rather than representing a wide range of outputs that match the actual data. The generator has a more demanding task, as it has to learn how to turn random noise into meaningful images that can fool the discriminator. On the other hand, the discriminator has a more straightforward role of distinguishing actual images from synthetic ones, which is why it is trained faster and usually starts to dominate the generator.

Given that training a GAN can take days, weeks, or even months, it is often inefficient to train a model from scratch. In order to speed up training, a transfer learning technique can be applied. This implies using a previously trained model as a starting point for a new task. Such models have already adopted useful features and representations that are necessary for image generation, which can significantly shorten training time and enable the faster achievement of quality results. The publicly available StyleGANv3 model, trained on diverse landscapes, was obtained from [44]. Landscapes share many similarities with collected satellite images, such as different terrain shapes, colors, and textures, which allow previously learned model features, such as edge detection and natural pattern recognition, to be successfully applied to satellite images during further training.

After applying the transfer learning technique, the discriminator gained too much dominance over the generator, causing the balance between them to collapse. In [45], the authors point out that the training process is considered complete when the generator generates images that are so realistic that the discriminator cannot reliably distinguish them from the real ones better than random guessing. This balance was achieved when the batch size was set to 32 samples, and the learning rate was set to 0.001 for the generator and 0.0001 for the discriminator to give the generator a slight advantage.

The StyleGANv3 model uses two additional parameters, *ticks* and *kimgs*, for transfer learning. *Kimgs* represents the number of kilo images per tick, where each *tick* corresponds to a fixed number of training iterations. In our case, the StyleGANv3 model was trained for 900 *ticks*, with four *kimgs* per *tick*, which amounts to approximately 3.6 million images processed during the training phase. Given that the training dataset consisted of 100,000 unique images, it can be inferred that each image was used an average of 36 times during the training process. Next, we present the assessment of the results of StyleGANv3 transfer learning in the form of generated image quality and the losses of generator and discriminator.

3.2.1. Generated Image Quality

While training a GAN, monitoring the progress and quality of the generated images is critical. In addition to visual inspection by the human eye, specific metrics can be used to evaluate the quality of GAN-generated images. For this purpose, we used Fréchet Inception Distance (FID) [46] for assessing the quality of images produced by the model. It is a valuable tool for determining the similarity between generated and real images. FID measures the similarity between the features of generated and actual images, and it is defined as

$$FID = \|\mu_a - \mu_g\|^2 + \mathrm{Tr}(\Sigma_a + \Sigma_g - 2(\Sigma_a\Sigma_g)^{1/2}), \tag{1}$$

where

$$\begin{aligned}
\mu_r &: \text{Mean of the feature representations of the real images;}\\
\mu_g &: \text{Mean of the feature representations of the generated images;}\\
\Sigma_r &: \text{Covariance matrix of the feature representations of the real images;}\\
\Sigma_g &: \text{Covariance matrix of the feature representations of the generated images;}\\
\|\mu_r - \mu_g\|^2 &: \text{Squared Euclidean distance between the means of the feature representations;}\\
\mathrm{Tr} &: \text{Trace of a matrix (sum of the diagonal elements);}\\
(\Sigma_r\Sigma_g)^{1/2} &: \text{Matrix square root of the product of the covariance matrices.}
\end{aligned}$$

Therefore, FID compares the means and covariances of extracted feature distributions, while lower FID scores indicate that the generated images are closer to the actual images.

We follow the feature extraction procedure described in [47]. The FID calculation begins with extracting features from generated and actual images using a pre-trained Inception v3 network, i.e., a convolutional neural network intended for image classification proposed in [48]. Then, the FID values are calculated based on extracted features according to Equation (1).

In Figure 4, the graph illustrates the fluctuation of the FiD throughout the StyleGAN training process. The x-axis represents *kimgs*, ranging from 0 to 3600, while the y-axis denotes the FID value. At the onset of training, the FID exhibits a notably high value of approximately 170, indicative of significant dissimilarities between the generated images and those in the dataset. Notably, applying transfer learning techniques leads to a steep decline in FID during the initial training phases, signifying rapid enhancement in the quality of the generated images. Subsequently, as the training progresses, the rate of decline in FID attenuates, exhibiting a consistent, gradual reduction with minimal oscillation. Ultimately, upon completing the 3600 *kimgs* training, the FID stabilizes at a value of less than 25, signaling a substantial improvement in the fidelity of the generated images.

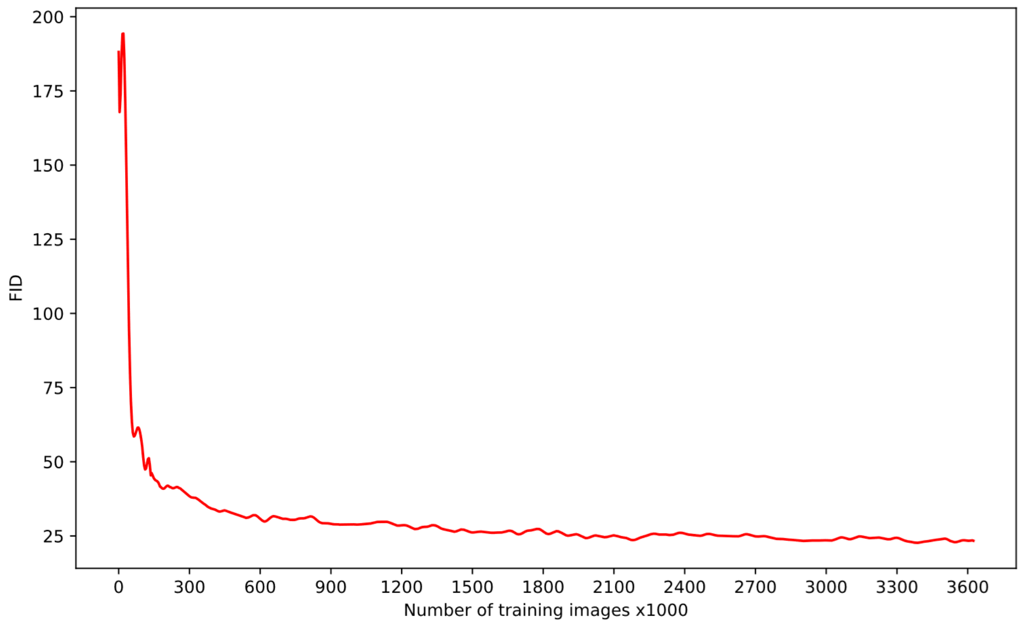


Figure 4. Change of FID values while training the model.

3.2.2. Generator and Discriminator Losses

The roles of generator and discriminator losses are pivotal in the training of GANs. The generator loss measures its ability to deceive the discriminator and generate authentic-looking images. The objective is to maximize the likelihood of the discriminator classifying the generated images as real while minimizing the generator loss throughout the training process. The losses of StyleGANv3 for discriminator and generator are defined in [21]. Conversely, discriminator loss signifies the discriminator's aptitude to differentiate between real and generated images. The discriminator aims to minimize its loss, indicating its tendency to accurately classify real images as real (resulting in low loss) and generated images as generated (resulting in high loss). A low overall loss for the discriminator indicates its successful discrimination of real and generated images.

In Figure 5, the fluctuations in loss values for both the generator (represented by the green line) and the discriminator (illustrated by the red line) during the training of the model are visualized. The x-axis denotes the number of *ticks* from 0 to 900, while the y-axis represents the corresponding loss values. At the outset of the training, both the generator's and the discriminator's loss values are notably high and exhibit oscillatory behavior. However, as the training progresses, the losses for both components gradually diminish and stabilize. Although oscillations persist, their amplitude decreases. No singular dominance between the two components is evident. It is worth noting that the discriminator consistently exhibits lower loss values compared to the generator, which is a common characteristic in GANs, reflecting the initial ease with which the discriminator can distinguish real from generated images.

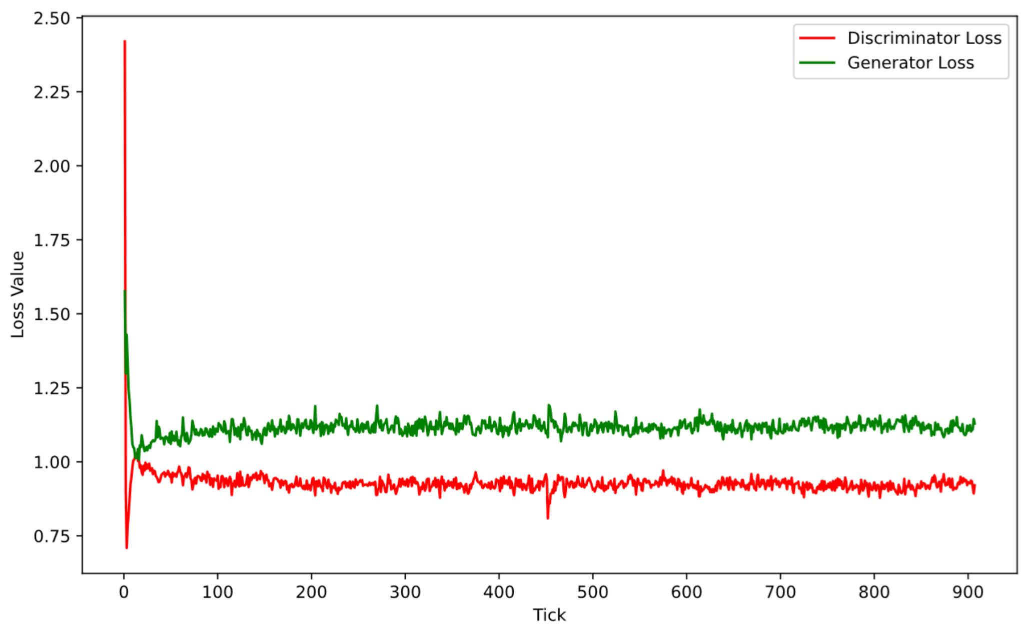


Figure 5. Change of loss during the training of the model.

3.3. Proposed Method

Here, we proposed a new CNN architecture for synthetic image detection. This approach is characterized by an intricate architecture that amalgamates two principal networks through a series of intermediate operations. Specifically, these networks comprise the convolutional layers, which are responsible for the initial processing and feature extraction from the images, and the fully connected layers, which play a critical role in the classification process based on the features identified. This refined framework is adept at capturing the intricacies and subtleties present within images, thereby facilitating a more nuanced and accurate classification outcome. The employment of a CNN in the image classification task exemplifies the confluence of advanced computational techniques and theoretical insights, heralding a more refined and practical approach to visual data analysis.

The architecture of the proposed CNN model is structured with multiple layers, each designed to methodically identify and analyze features from the input images, ultimately leading to a fully connected layer for the classification task. This architecture is displayed in Figure 6.

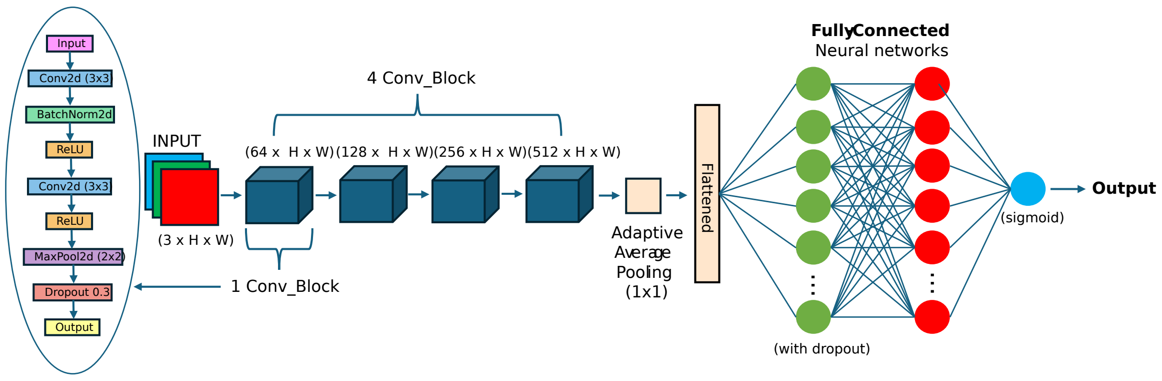


Figure 6. Architecture of the proposed methodology based on a CNN.

The input layer accepts images of dimensions 3 × H × W, indicating three color channels across height and width. At the same time, each convolutional block in the architecture is comprised of a sequence that includes two convolutional layers followed by batch normalization and the application of ReLU activation functions. The initial convolutional layer within each block employs a 3 × 3 kernel for feature extraction. The incorporation of batch normalization aids in stabilizing the training process and expediting convergence, while the ReLU activation function introduces non-linearity, allowing for the capture of complex patterns. Furthermore, since batch normalization is computationally intensive [49] and the proposed CNN architecture is lightweight, using only one such operation per individual convolutional block was sufficient to ensure stability training, as demonstrated in Section 4. Max-pooling layers with a 2 × 2 window are used to reduce the feature maps both in spatial dimensions and computational demand. Additionally, a dropout layer with a 0.3 rate is applied to overcome overfitting, randomly disabling a portion of the neurons during the training phase.

Features are extracted in the network through four convolutional blocks, progressively increasing the number of filters (64, 128, 256, and 512) to capture more abstract and higher-level features at deeper layers. An Adaptive Average Pooling layer follows the final convolutional block, reducing each feature map to a size of 1 × 1, thus converting the spatial dimensions into a single vector representation. This vector is then input into a fully connected neural network comprising two hidden layers, each with 512 neurons. To further mitigate overfitting and boost the network's ability to generalize, dropout is applied between the fully connected layers. The output layer employs a sigmoid activation function suited for binary classification tasks by producing a probability score for each class.

4. Results and Discussion

4.1. Validation Setup

The proposed method was implemented in the Python programming language on the Microsoft Windows 11 operating system, while all conducted experiments were performed on a workstation with AMD Ryzen 7 3800 × CPU and 32 GB of main memory. In order to ensure the reproducibility of the experiments, the proposed method relied on the PyTorch machine learning library. In this paper, 70% of the data were used for training the model, 15% for the validation set, and the remaining 15% for the test set. Sample images were selected randomly to ensure the unbiasedness and representativeness of each set.

In addition to generated satellite imagery, two benchmark datasets were also used to validate the proposed method. These were CIFAKE [50] and Midjourney v6 [51]. As presented in Section 3.1, we created a synthetic dataset of 100,000 images, where 50,000 images were generated with StyleGANv3 and concatenated with 50,000 real ones for training the StyleGANv3. CIFAKE, on the other hand, is a publicly available dataset containing 60,000 synthetic and 60,000 real images. The real images comprise the CIFAR-10 dataset, while the synthetic images were generated using the Stable Diffusion 1.4 model [50] and scaled to 32 × 32 pixels. Similarly, Midjourney v6 [51] consists of a total of 50,000 images, 12,500 of which were synthetic images generated by the authors, and the remaining 12,500 were real images collected from public datasets by the authors and scaled to 64 × 64 pixels. The remaining 25,000 images are created by image transformations such as vertical and horizontal mirroring and rotation.

The data allocation into training, validation, and test subsets for each dataset, together with image sizes, is presented in Table 1. All three datasets used in this paper contain an equal number of real and AI-generated images, thus ensuring data balance.

Table 1. Summary of datasets.

Dataset	Images Count	Train Set	Validation Set	Test Set	Image Size
CIFAKE	120,000	84,000	18,000	18,000	32 × 32
Midjourney v6	50,000	35,000	7500	7500	64 × 64
StyleGANv3	100,000	70,000	15,000	15,000	128 × 128

The evaluation of the proposed method on these datasets is based on the confusion matrix [52], with the following metrics to assess the performance of learned classification models:

- precision, defined as $Precision = \frac{TP}{TP+FP}$;
- recall, defined as $Recall = \frac{TP}{TP+FN}$;
- accuracy, defined as $Accuracy = \frac{TP+TN}{\text{all classified samples}}$; and
- F1 score, defined as $F1score = 2 * \frac{precission*recall}{precission+recall}$,

where TP denotes the number of correctly classified positive examples, TN is the number of correctly classified negative examples, FP denotes the number of incorrectly classified positive examples, and FN is the number of incorrectly classified negative examples.

4.2. Performance Evaluation

In order to assess the actual performance and generalization ability, the proposed method was evaluated on a separate test dataset. The test set, which is 15% of the total dataset, set apart before starting training, consists of unseen images that serve as a final assessment of the network's ability to classify unknown examples accurately. The models were trained and tested using different random data for each of the 10 training sessions. The images were randomly selected for training, testing, and validation for each training session. This ensured that, in every session, different subsets of images were used for training, validation, and testing purposes. We set the number of training epochs to ten for the Midjourney v6 and StyleGANv3 in order to ensure stable convergence without encountering issues like overfitting or underfitting as, at the 11th or 12th epoch, the training loss continues to decrease, but the validation loss increases and the overall accuracy does not improve on the test set. This was noted also for CIFAKE after the 15th epoch. For the test sets, we averaged accuracies from all 10 training sessions and calculated the associated standard deviations. The results are presented in Table 2. On the other hand, loss and accuracy curves for all three datasets are displayed in Figures 7–9 from a training session that yielded the highest accuracy on the test set.

Table 2. Average evaluation results in [%] on the test sets for selected datasets.

Dataset	Accuracy	Precision	Recall	F1 Score	Epochs
CIFAKE	96.60 ± 0.26	96.66 ± 0.82	96.53 ± 0.98	96.59 ± 0.28	15
Midjourney v6	99.94 ± 0.18	99.63 ± 0.22	99.79 ± 0.07	99.20 ± 0.08	10
StyleGANv3	98.69 ± 0.18	98.31 ± 0.42	99.11 ± 0.33	98.70 ± 0.18	10

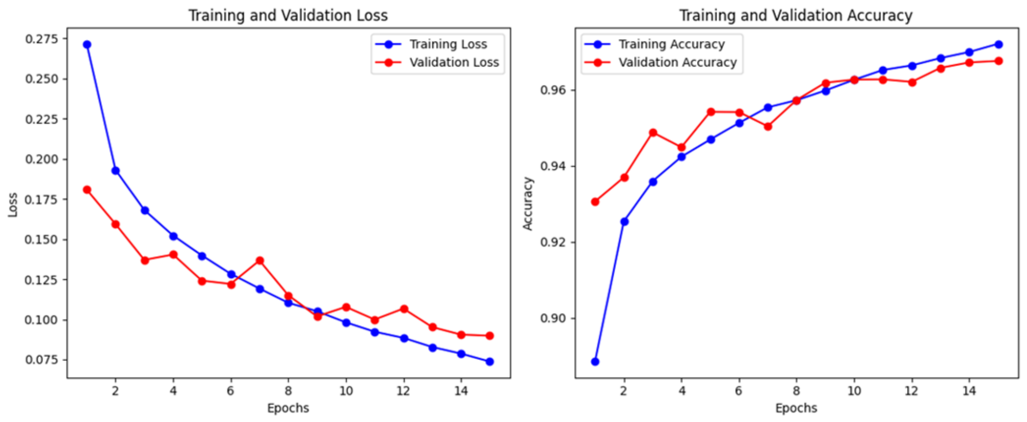


Figure 7. Loss and accuracy curves for the CIFAKE dataset (training and validation sets).

Figure 7 shows the loss versus accuracy curves of the model on the CIFAKE dataset. The losses on the training and validation sets decrease as the number of epochs increases, indicating the successful learning and generalization of the model. The accuracy on both sets increases, reaching high values after a few epochs, while the learning process is relatively stable and has minimal divergence. However, in the first eight epochs, the accuracy on the validation set is higher than that on the training set. This is likely due to the relatively high dropout rate of 0.3 that is active during training [11]. As training progresses, the accuracy of the training set surpasses that of the validation set. This indicates that the model fits the training data more closely, including any noise or anomalies introduced in the training set. On the testing sets of the CIFAKE dataset, the model achieves an average accuracy of 96.60%, precision of 96.66%, and recall of 96.53%. These high values indicate that the model correctly classifies a large percentage of images, with very few false positive and false negative examples. The average precision of 96.66% indicates the high ability of the model to detect AI-generated images as fake correctly, while the recall of 96.53% shows the model's capacity to identify 96.53% of instances of fake images, ensuring that only 3.47% of fakes are undetected. A high F1 score of 96.59% confirms the balance between precision and recall.

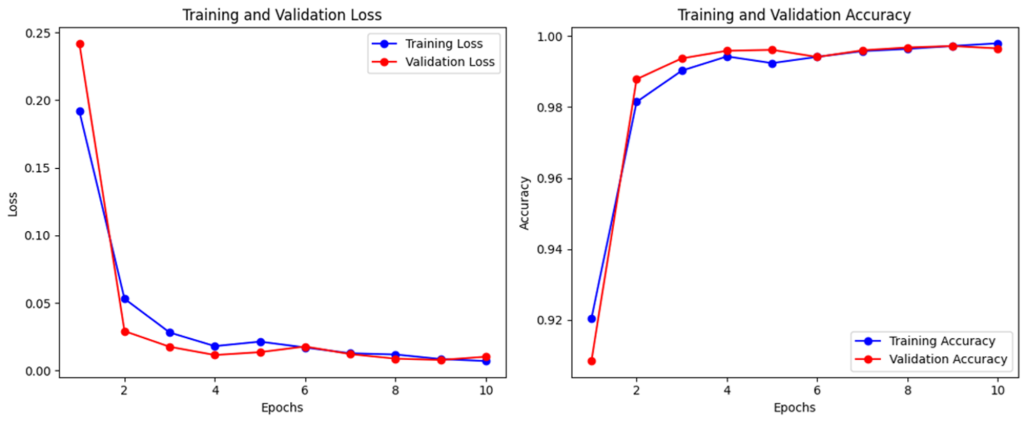


Figure 8. Loss and accuracy curves for the Midjourney v6 dataset (training and validation sets).

Similar results can be seen for the Midjourney v6 dataset. In the first six epochs, the accuracy on the validation set is higher than that on the training set, while, as training progresses, the accuracy of the training set surpasses that of the validation set. The loss and accuracy curves for the model on the Midjourney v6 dataset display a significant decrease in loss on the training and validation sets during the first few epochs, after which a slight convergence occurs (see Figure 8). In contrast, the model's accuracy on the training and validation data shows a sharp increase in the initial epochs, after which it stabilizes and continues to increase at a lower rate. The model gradually achieves very high average results on the test sets with an accuracy of 99.94%, a precision of 99.63%, a recall of 99.79%, and an F1 measure of 99.20%. A recall close to 100% suggests that the model almost perfectly identifies all real images.

Interestingly, on the StyleGANv3 dataset, the validation set achieves better performance than the training set, as shown in Figure 9. This phenomenon can be attributed to the application of regularization techniques such as dropout layers in the network architecture. Dropout layers randomly turn off a certain percentage of neurons during the training process, thereby reducing the possibility of overfitting the model to the training data and improving the ability to generalize to unseen examples [11]. When we train it for another five epochs, the accuracy of the training set surpasses that of the validation set, which starts to decrease, indicating that 10 epochs is sufficient for optimal training. Moreover, the average results achieved on the test sets were 98.69% accuracy, 98.31% precision, 99.11% recall, and an F1 measure of 98.70%. Both precision and recall are very high and balanced, indicating that the model makes very few false positive (FP) and false negative (FN) errors. In other words, when the model predicts that an image is AI-generated, it is correct in 98.31% of the samples.

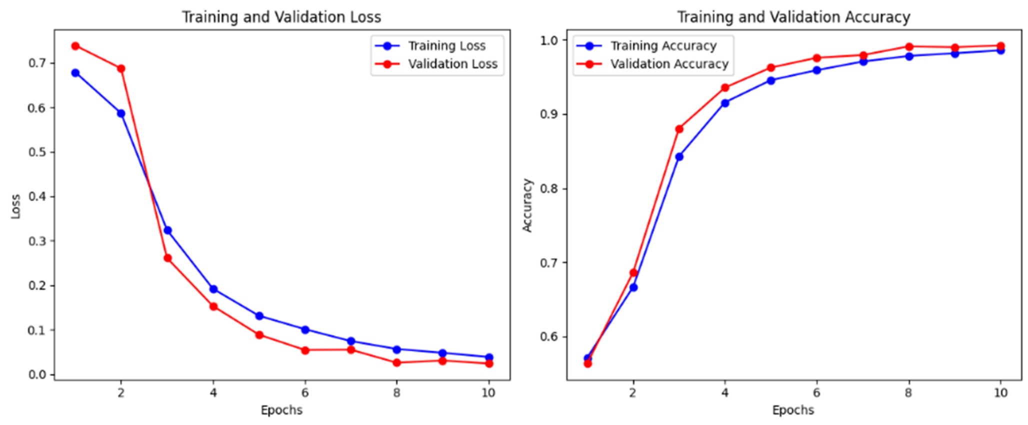


Figure 9. Loss and accuracy curves for the StyleGANv3-generated dataset (training and validation sets).

As we generated a dataset with StyleGANv3 and results for Midjourney v6 have not been available in published research since it was only released at the end of the previous year, we compared the proposed method on the CIFAKE dataset with five state-of-the-art methods. The comparison includes CIFAKE CNN [50], Dual-Input Neural Model (DINM) [32], while the results from the ResNet [15], VGGNet [13], and DenseNet [14] are obtained from [53]. Table 3 summarizes the results.

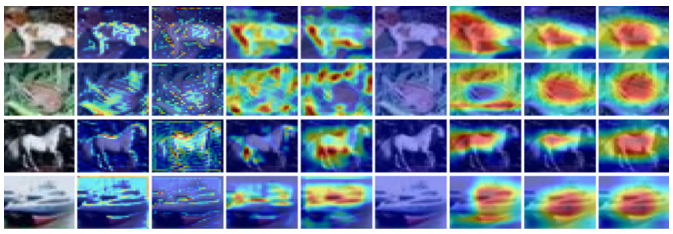
Table 3. Highest accuracy in [%] reported on the test set for the CIFAKE dataset.

Method	Accuracy
The proposed method	97.32
CIFAKE CNN	94.80
DINM	94.51
ResNet	94.95
VGGNet	96.00
DenseNet	97.74

While DenseNet set the benchmark by attaining the highest level of accuracy, reaching 97.74%, and thus emerged as the most effective model among those evaluated, the proposed method is just a little behind, showcasing commendable performance at 97.32%. VGGNet also demonstrated strong capabilities, achieving an accuracy of 96.00%. In comparison, CIFAKE CNN, DINM, and ResNet reported somewhat lower accuracies, with values of 94.80%, 94.51%, and 94.95%, respectively. In summary, both DenseNet and the proposed methods surpassed other evaluated approaches in terms of effectiveness when applied to the CIFAKE dataset. This overall performance comparison suggests that the proposed method holds significant promise for future research and applications within this domain, marking a noteworthy advancement in the field.

4.3. Explaining the Decisions of the Proposed Method

Finally, we discuss the explanations of the proposed method decisions using Grad-CAM++ (v.1.2.4), the approach described in [42]. Explanations were provided for all three datasets used in this research, while some samples of real and fake images used for explanation were randomly selected. As discussed in Section 2.3, Grad-CAM++ creates a heatmap to pinpoint the importance of different image regions that influence the decision-making process, where warmer colors indicate greater importance for a specific class and cooler colors are less important. Moreover, Figures 10–12 show Grad-CAM++ activations for each of the eight convolutional layers of the proposed network architecture. In those figures, examples (a) depict activations for real images, while examples (b) show synthetic images. Each row starts with the original image without the heat map, followed by subsequent images showing the heat maps of the activations for a specific convolutional network layer. These visualizations are from the last epoch of model training and illustrate the model's ability to distinguish between real and synthetic images.


(**a**)

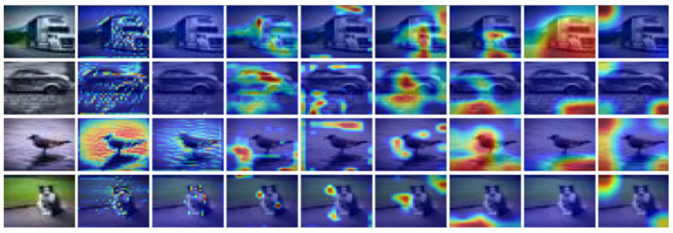
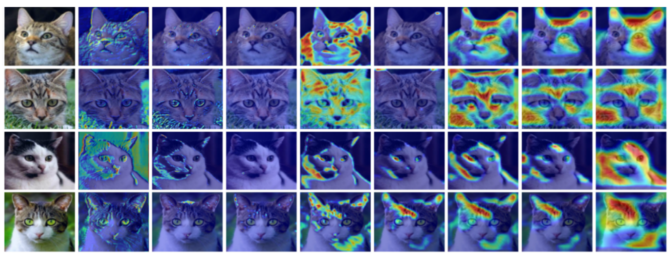

(**b**)

Figure 10. Examples in (**a**) illustrate activations for real images, while examples in (**b**) showcase activations for synthetic images from the CIFAKE dataset.


(**a**)

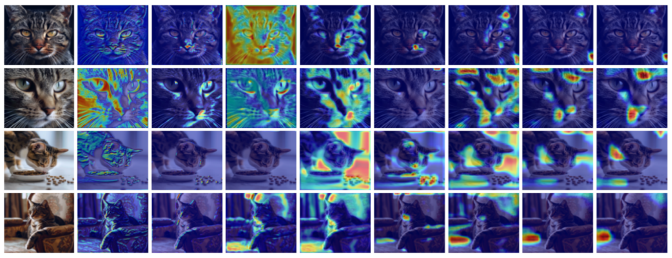
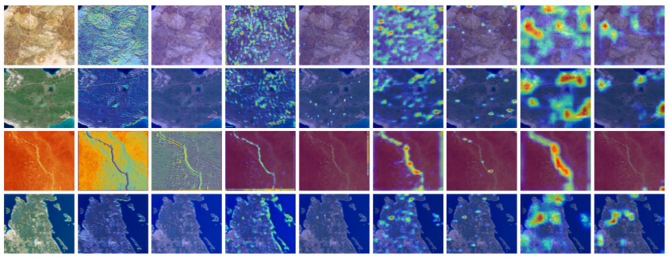

(**b**)

Figure 11. Examples in (**a**) illustrate activations for real images, while examples in (**b**) showcase activations for synthetic images from the Midjourney v6 dataset.


(**a**)

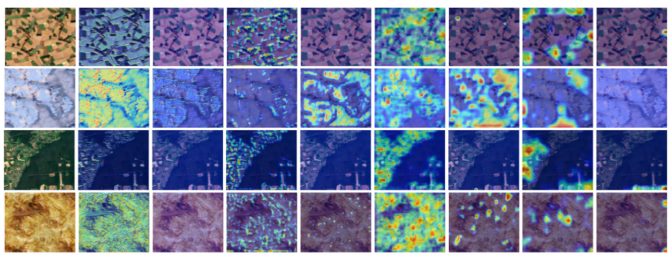

(**b**)

Figure 12. Examples in (**a**) illustrate activations for real satellite images, while examples in (**b**) showcase activations for synthetic images generated using the StyleGANv3 (as described in Section 3.2).

As can be seen from Figure 10, the main objects in real images predominantly contribute to the recognition of the real images, while activations for synthetic images are more diffused and usually do not focus on the image object itself but on small visual irregularities in the background. This is also aligned with the previous research, for example, in [50]. We observe similar patterns for the Grad-CAM++ activations in Figures 11 and 12. In the initial convolutional layers, the activations are small and spread across the image, indicating that the network initially detects general visual features in different parts of the image. However, as we progress through the network layers, the activations become more focused and concentrated in specific areas of the real image. The observation suggests a distinctive difference in how activations behave in real versus synthetic images. In real images, activations focus on the main object, presumably due to their clear, central role in the composition. Conversely, synthetic images show activations clustering not around the primary object but around the peripheries, including object edges, backgrounds, or areas with complex textures. This indicates a fundamental divergence in how real and synthetic images are processed or represented, possibly due to the inherent differences in their composition and the elements that are more pronounced or defining in each type.

Generally, we notice that, when the network converges to the optimal solution, the heatmaps of activations provide more useful insight into how the neural network learns through different layers and training epochs. Convergence is manifested in the reduction of the magnitudes of gradients and activations in later epochs, indicating that the network has successfully learned the essential features and patterns from the training data. Activations become smaller and more precise, and those that contribute the most to the classification are usually found in the last layers of the network, where very abstract and complex information is extracted.

On the one hand, the differences in activation patterns between real and synthetic images can serve as a guide for the next generation of architectures for generative models and advanced training techniques for this purpose. Furthermore, in the face of the growing prevalence of deepfakes and AI-generated media, these discoveries can play a vital role in the development of tools to counter misinformation. An in-depth comprehension of the variations in activation patterns at the AI-synthesized image level can provide substantial ability to platforms and regulators to identify and effectively mitigate the dissemination of fake content.

5. Conclusions

Our research demonstrates a substantial advancement in application, especially through the use of CNN architectures, to address the complex challenges in digital image recognition and generation. By creating a high-performance CNN-based method, we not only improve human ability to identify AI-generated images with greater accuracy but also, by using explanations, move closer to understanding the subtle processes of synthetic image recognition, thus bolstering the reliability and accuracy of digital data. The proposed method holds great promise for future research and applications in this field, as its accuracy is slightly below that of the best-performing method. Finally, explanations of the models indicate that activations in real images are often focused on the main object, while, in synthetic images, activations tend to be clustered around the edges of objects, in the background, or in areas with complex textures.

Author Contributions: Conceptualization, D.V. and T.K.; methodology, A.L.L. and D.V.; software, A.L.L. and T.K.; validation, A.L.L., R.K. and D.V.; formal analysis, D.V. and R.K.; investigation, D.V. and T.K.; resources, A.L.L.; data curation, A.L.L. and R.K.; writing—original draft preparation, A.L.L. and D.V.; writing—review and editing, R.K. and T.K.; visualization, A.L.L. and R.K.; supervision, T.K. and D.V.; project administration, T.K. and D.V.; funding acquisition, T.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data generation process is reported in this study, while other used data are available online.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

GAN	Generative Adversarial Network
CNN	Convolutional Neural Network
GPU	Graphics Processing Unit
FID	Fréchet Inception Distance
CPU	Central Processing Unit

References

1. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative Adversarial Networks. *Adv. Neural Inf. Process. Syst.* **2014**, *3*, 139–144. [CrossRef]
2. Masood, M.; Nawaz, M.; Malik, K.; Javed, A.; Irtaza, A.; Malik, H. Deepfakes generation and detection: State-of-the-art, open challenges, countermeasures, and way forward. *Appl. Intell.* **2022**, *53*, 3974–4026. [CrossRef]
3. Mirsky, Y.; Lee, W. The Creation and Detection of Deepfakes: A Survey. *ACM Comput. Surv.* **2021**, *54*, 1–41. [CrossRef]
4. Thakur, R.; Rohilla, R. Recent advances in digital image manipulation detection techniques: A brief review. *Forensic Sci. Int.* **2020**, *312*, 110311. [CrossRef]
5. Fridrich, J.; Soukal, D.; Lukás, J. Detection of Copy-Move Forgery in Digital Images. *Int. J. Comput. Sci. Issues* **2003**, *3*, 55–61.
6. Li, G.; Wu, Q.; Tu, D.; Sun, S. A Sorted Neighborhood Approach for Detecting Duplicated Regions in Image Forgeries Based on DWT and SVD. In Proceedings of the 2007 IEEE International Conference on Multimedia and Expo, Beijing, China, 2–5 July 2007; pp. 1750–1753. [CrossRef]
7. Bayram, S.; Sencar, T.; Memon, N. An efficient and robust method for detecting copy-move forgery. In Proceedings of the 2009 IEEE International Conference on Acoustics, Speech and Signal Processing, Taipei, Taiwan, 19–24 April 2009; pp. 1053–1056. [CrossRef]
8. Ferrara, P.; Bianchi, T.; De Rosa, A.; Piva, A. Image Forgery Localization via Fine-Grained Analysis of CFA Artifacts. *IEEE Trans. Inf. Forensics Secur.* **2012**, *7*, 1566–1577. [CrossRef]
9. He, Z.; Lu, W.; Sun, W.; Huang, J. Digital image splicing detection based on Markov features in DCT and DWT domain. *Pattern Recognit.* **2012**, *45*, 4292–4299. [CrossRef]
10. Sharma, D.K.; Singh, B.; Agarwal, S.; Garg, L.; Kim, C.; Jung, K.H. A Survey of Detection and Mitigation for Fake Images on Social Media Platforms. *Appl. Sci.* **2023**, *13*, 10980. [CrossRef]
11. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016; p. 800. Available online: http://www.deeplearningbook.org (accessed on 18 July 2024)..
12. Bianchini, M.; Scarselli, F. On the Complexity of Neural Network Classifiers: A Comparison between Shallow and Deep Architectures. *IEEE Trans. Neural Netw. Learn. Syst.* **2014**, *25*, 1553–1565. [CrossRef]
13. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
14. Huang, G.; Liu, Z.; Van Der Maaten, L.; Weinberger, K.Q. Densely connected convolutional networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 4700–4708. [CrossRef]
15. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778. [CrossRef]
16. Gu, J.; Wang, Z.; Kuen, J.; Ma, L.; Shahroudy, A.; Shuai, B.; Liu, T.; Wang, X.; Wang, G.; Cai, J.; et al. Recent advances in convolutional neural networks. *Pattern Recognit.* **2018**, *77*, 354–377. [CrossRef]
17. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.C. Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 4510–4520.
18. Abbas, M.N.; Ansari, M.S.; Asghar, M.N.; Kanwal, N.; O'Neill, T.; Lee, B. Lightweight deep learning model for detection of copy-move image forgery with post-processed attacks. In Proceedings of the 2021 IEEE 19th World Symposium on Applied Machine Intelligence And Informatics (SAMI), Herl'any, Slovakia, 21–23 January 2021; pp. 125–130.
19. Karras, T.; Laine, S.; Aila, T. A Style-Based Generator Architecture for Generative Adversarial Networks. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 4396–4405. [CrossRef]
20. Karras, T.; Laine, S.; Aittala, M.; Hellsten, J.; Lehtinen, J.; Aila, T. Analyzing and Improving the Image Quality of StyleGAN. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 8107–8116. [CrossRef]
21. Karras, T.; Aittala, M.; Laine, S.; Härkönen, E.; Hellsten, J.; Lehtinen, J.; Aila, T. Alias-Free Generative Adversarial Networks. In Proceedings of the Neural Information Processing Systems, Online, 6–14 December 2021. [CrossRef]
22. Pei, S.; Da Xu, R.Y.; Xiang, S.; Meng, G. Alleviating mode collapse in GAN via diversity penalty module. *arXiv* **2021**, arXiv:2108.02353.
23. Ghosh, A.; Kulharia, V.; Namboodiri, V.; Torr, P.H.; Dokania, P.K. Multi-agent Diverse Generative Adversarial Networks. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 8513–8521. [CrossRef]
24. Liu, H.; Li, B.; Wu, H.; Liang, H.; Huang, Y.; Li, Y.; Ghanem, B.; Zheng, Y. Combating mode collapse via offline manifold entropy estimation. *Aaai Conf. Artif. Intell.* **2023**, *37*, 8834–8842. [CrossRef]
25. Radford, A.; Metz, L.; Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv* **2015**, arXiv:1511.06434.

26. Rombach, R.; Blattmann, A.; Lorenz, D.; Esser, P.; Ommer, B. High-Resolution Image Synthesis with Latent Diffusion Models. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022; pp. 10674–10685. [CrossRef]
27. LeCun, Y.; Boser, B.; Denker, J.S.; Henderson, D.; Howard, R.E.; Hubbard, W.; Jackel, L.D. Backpropagation Applied to Handwritten Zip Code Recognition. *Neural Comput.* **1989**, *1*, 541–551. [CrossRef]
28. Keeler, J.; Rumelhart, D.; Leow, W. Integrated segmentation and recognition of hand-printed numerals. *Adv. Neural Inf. Process. Syst.* **1990**, *3*, 557–563.
29. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet classification with deep convolutional neural networks. *Commun. ACM* **2017**, *60*, 84–90. [CrossRef]
30. Bayar, B.; Stamm, M.C. A deep learning approach to universal image manipulation detection using a new convolutional layer. In Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security, Vigo Galicia, Spain, 20–22 June 2016; pp. 5–10. [CrossRef]
31. Kaur, R.; Kumar, R.; Gupta, M. Review on Transfer Learning for Convolutional Neural Network. In Proceedings of the 2021 3rd International Conference on Advances in Computing, Communication Control and Networking (ICAC3N), Greater Noida, India, 17–18 December 2021; pp. 922–926. [CrossRef]
32. Gallagher, J.; Pugsley, W. Development of a Dual-Input Neural Model for Detecting AI-Generated Imagery. *arXiv* **2024**, arXiv:2406.13688.
33. Azad, M.; Chikalov, I.; Moshkov, M. Representation of Knowledge by Decision Trees for Decision Tables with Multiple Decisions. *Procedia Comput. Sci.* **2020**, *176*, 653–659. [CrossRef]
34. Chandler, R. On the use of generalized linear models for interpreting climate variability. *Environmetrics* **2005**, *16*, 699–715. [CrossRef]
35. Carvalho, D.V.; Pereira, E.M.; Cardoso, J.S. Machine Learning Interpretability: A Survey on Methods and Metrics. *Electronics* **2019**, *8*, 832. [CrossRef]
36. Bengio, Y.; Courville, A.; Vincent, P. Representation Learning: A Review and New Perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.* **2013**, *35*, 1798–1828. [CrossRef] [PubMed]
37. Ribeiro, M.T.; Singh, S.; Guestrin, C. Why Should I Trust You? Explaining the Predictions of Any Classifier. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 1135–1144. [CrossRef]
38. Štrumbelj, E.; Kononenko, I. Explaining prediction models and individual predictions with feature contributions. *Knowl. Inf. Syst.* **2013**, *41*, 647–665. [CrossRef]
39. Slack, D.; Hilgard, S.; Jia, E.; Singh, S.; Lakkaraju, H. Fooling LIME and SHAP: Adversarial Attacks on Post hoc Explanation Methods. In Proceedings of the AIES '20: AAAI/ACM Conference on AI, Ethics, and Society, New York, NY, USA, 7–9 February 2020; pp. 180–186. [CrossRef]
40. Simonyan, K.; Vedaldi, A.; Zisserman, A. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv* **2013**, arXiv:1312.6034.
41. Zeiler, M.D.; Fergus, R. Visualizing and understanding convolutional networks. In Proceedings of the Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, 6–12 September 2014; pp. 818–833.
42. Chattopadhay, A.; Sarkar, A.; Howlader, P.; Balasubramanian, V.N. Grad-CAM++: Generalized Gradient-Based Visual Explanations for Deep Convolutional Networks. In Proceedings of the 2018 IEEE Winter Conference on Applications of Computer Vision (WACV), Lake Tahoe, NV, USA, 12–15 March 2018; pp. 839–847. [CrossRef]
43. Copernicus. Copernicus in Detail. Available online: https://www.copernicus.eu/en/about-copernicus/copernicus-detail (accessed on 18 July 2024).
44. Pinkney, J.N.M. Awesome Pretrained StyleGAN3. 2021. Available online: https://github.com/justinpinkney/awesome-pretrained-stylegan3 (accessed on 18 July 2024).
45. Bok, V.; Langr, J. *GANs in Action: Deep Learning with Generative Adversarial Networks*, 1st ed.; Manning: Shelter Island, NY, USA, 2019.
46. Obukhov, A.; Krasnyanskiy, M. Quality Assessment Method for GAN Based on Modified Metrics Inception Score and Fréchet Inception Distance. In *Software Engineering Perspectives in Intelligent Systems*; Silhavy, R., Silhavy, P., Prokopova, Z., Eds.; Springer: Berlin/Heidelberg, Germany, 2020; pp. 102–114. [CrossRef]
47. Heusel, M.; Ramsauer, H.; Unterthiner, T.; Nessler, B.; Hochreiter, S. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. In Proceedings of the Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017.
48. Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; Wojna, Z. Rethinking the Inception Architecture for Computer Vision. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016. [CrossRef]
49. Zhu, Y.; Du, J.; Zhu, Y.; Wang, Y.; Ou, Z.; Feng, F.; Tang, J. Training BatchNorm Only in Neural Architecture Search and Beyond. *arXiv* **2021**, arXiv:2112.00265.
50. Bird, J.J.; Lotfi, A. CIFAKE: Image Classification and Explainable Identification of AI-Generated Synthetic Images. *IEEE Access* **2024**, *12*, 15642–15650. [CrossRef]
51. Midjourney Inc. Midjourney. Available online: https://www.midjourney.com (accessed on 18 July 2024).

52. Han, J.; Kamber, M.; Pei, J. *Data Mining: Concepts and Techniques*, 3rd ed.; Morgan Kaufmann Publishers Inc.: San Francisco, CA, USA, 2011.
53. Wang, Y.; Hao, Y.; Cong, A.X. Harnessing Machine Learning for Discerning AI-Generated Synthetic Images. *arXiv* **2024**, arXiv:2401.07358.