applied sciences

MDPI

Article

Advancing GAN Deepfake Detection: Mixed Datasets and Comprehensive Artifact Analysis

Tamer Say [1], Mustafa Alkan [2,*] and Aynur Kocak [2]

[1] Department of Information Security Engineering, Graduate School of Natural and Applied Sciences, Gazi University, 06560 Ankara, Turkey; tamer.say1@gazi.edu.tr
[2] Department of Electrical and Electronics Engineering, Faculty of Technology, Gazi University, 06560 Ankara, Turkey; aynurkocak@gazi.edu.tr
* Correspondence: alkan@gazi.edu.tr

Abstract: The rapid advancement of synthetic media, while beneficial, has also spawned GAN-generated deepfakes, which pose risks, including misinformation and digital fraud. This paper investigates the detectability of GAN-generated static images, focusing on residual artifacts that are imperceptible to humans but detectable through digital analysis. Our approach introduces three key advancements: (1) a taxonomy for classifying GAN residues in deepfake detection; (2) a unique mixed dataset combining StyleGAN3, ProGAN, and InterfaceGAN to aid cross-model detection research; and (3) a combination of frequency space analysis and RGB color correlation methods to improve artifact detection. Covering three different transform methods, three GAN models, and twelve classification methods, ours is the most comprehensive study of detection of static deepfake face images produced by GANs. Our results demonstrate that artifact-based detection can achieve high accuracy, precision, recall, and F1 scores, challenging prior assumptions about the detectability of synthetic face images.

Keywords: computer vision; deepfakes; synthetic media; machine learning

Received: 30 November 2024
Revised: 11 January 2025
Accepted: 15 January 2025
Published: 18 January 2025

Citation: Say, T.; Alkan, M.; Kocak, A. Advancing GAN Deepfake Detection: Mixed Datasets and Comprehensive Artifact Analysis. *Appl. Sci.* **2025**, *15*, 923. https://doi.org/10.3390/app15020923

1. Introduction

Technological advancements in the past decade have accelerated the rate and quality of synthetic media being produced. This technology has various beneficial applications in diverse fields. In medical diagnostics [1], applications have been used to detect lung changes from COVID-19, and for brain tumor detection by SeGAN. Heavy industrial applications of GANs include flaw detection in materials, with studies on various surfaces using DCGAN and WGAN. Visual synthesis is another significant area, with models like Stack GAN being used for generating images from text and Discovery GAN for style transfer. Synthetic media have also rapidly been adopted for creative expression. Recent studies have employed GANs for restoring damaged artworks, achieving success in reconstructing less-damaged mosaics.

As with any revolutionary technology, the negative implications of synthetic media are significant, posing risks such as violations of individual rights, the spread of misinformation, and digital fraud. This paper investigates the detectability of visual forgery. In particular, it focuses on determining authenticity of static image content generated by Generative Adversarial Networks (GANs), with emphasis on residual artifacts that elude human perception but can be detected through digital analysis.

GAN models [2] excel at producing hyper-realistic synthetic images, popularly known as "deepfakes", due to their architecture, which continually improves the Generator neu-

ral network until the Discriminator neural network can no longer differentiate between generated and real data [3]. Recent advancements in training techniques and data quality make it increasingly difficult to distinguish GAN-generated images from authentic ones. Despite these difficulties, the literature indicates that static images generated by GANs retain digital imprints of their origins, which can assist in detection.

However, detection of GAN deepfakes remains a challenging problem due to the following reasons: (i) These digital artifacts, also known as residues or residuals, are created during the learning phase across various layers of the GANs, which makes these artifacts challenging to identify. In particular, differences in training and testing conditions make it difficult to apply detection methods in real-world scenarios. (ii) While many existing studies concentrate on using datasets from known Generative Adversarial Networks (GANs), identifying high-resolution deepfakes generated by unseen models remains particularly challenging. (iii) The literature lacks a systematic classification and study of residues. (iv) The implementation of noise-canceling methods [4] and residue removal methods [5–7] in advanced GAN models significantly hampers detectability. Studies have shown that specific GAN models, like PGGAN, can produce detectable artifacts, while others, like STGAN, do not synthesize noise values, complicating detection.

In this research, we present a GAN simulation capable of detecting artifacts and deepfakes by identifying common features across different applications that examine GAN-generated artifacts. This simulation offers a multi-faceted method for detecting artifacts from specific GAN models, with results shared and analyzed for detection and classification performance in the context of deepfake identification. This research introduces four significant advancements:

1. Proposed Taxonomy: A classification framework for the study of artifacts in GAN deepfake detection, particularly in facial image deepfakes, is proposed. A major problem this paper addresses is the absence of classification studies defining GAN residues. By analyzing GAN residues, this study contributes to a deeper understanding of detection methods and presents a taxonomy for artifact detection. In addition, a detection study was conducted using a unique facial image dataset crafted from three identified GAN models, allowing for a comparative analysis of model performance and assessing the model's ability to recognize images produced by other unseen models.
2. Development of a Unique Multi-GAN Dataset: Unlike many studies that focus on single known GANs, our approach combines three different GAN models, StyleGAN3, ProGAN and InterfaceGAN [8–10]. These GAN models were particularly selected for their popularity and success in the generation of deepfake face images. We created a mixed dataset to aid researchers in exploring GAN detection, particularly for facial deepfakes, addressing the challenges associated with the absence of diverse datasets in the field. Additionally, our specialized mixed dataset for GAN detection will serve as a valuable resource for researchers, facilitating studies involving mixed datasets generated by various GAN models—a notable gap in the current landscape.
3. Combining Frequency Space and RGB Color Correlation Methods of Artifact Detection: In particular, we analyze GAN definitions through frequency and generalization analyses of the identified residues and their repetition across various GAN models. By examining frequency space analyses and integrating them with additional residue detection methods, we aim to enhance the accuracy and performance of detection techniques in this field.
4. Our novel method achieves an accuracy of at least 87% as seen in Section 6, outperforming current methods such as the RealForensics detector [11], which has a state-of-the-art generalization performance on unseen datasets with a high accuracy

of 83% and 75% for real and fake images. In addition, our proposed method also achieves 100% accuracy when the classifier used is Support Vector Machine with RBF kernel.

In particular, we showcase that, contrary to previous belief [12], artifact-based synthetic face image detection can, in fact, produce extremely high accuracy, precision, recall, and F1 scores using various classifiers.

The purpose of this study is to enhance detection capabilities and achieve higher accuracy in identifying GAN-generated deepfakes of facial images. In particular, we wish to address the generalization problem in deepfake detection whereby a detector does well when the training and test data are created using the same techniques, but performs poorly when deepfakes from unfamiliar sources are presented to the detector, as is bound to happen in real-world scenarios [13]. By targeting the detection of unfamiliar deepfakes, we aim to analyze digital residues through a combination of feature extraction and frequency domain transformations. We choose this approach for two reasons. (i) Speed and efficiency of resource utilization: While several deepfake detection pipelines based on deep learning exist, which can automatically extract features and classify them, we choose to preprocess our images first for feature extraction and then follow this with classification with traditional machine learning classifiers because our approach is far more economical in terms of compute power usage and time taken. Our algorithm does not need training and can be directly applied on images from never before seen models. As seen in Section 6, our algorithm runs in a fraction of seconds.

(ii) Explainability: While previous deepfake detection applications have attempted image preprocessing, we have conducted a more in-depth exploration of feature extraction and adapted the features to better align with their detection capabilities when anomalous. Additionally, we addressed a critical gap in the field by introducing a multi-GAN dataset. The detection framework operates without prior knowledge of the multi-GAN datasets or their architectures; it relies solely on feature extraction and analysis to classify images as either authentic or synthetic, enabling further advancements in the field. The proposed detection framework integrates image preprocessing in the RGB color domain with analysis in the frequency spectrum, thereby achieving effective classification of images.

2. Related Work

We introduce deepfake generation and detection methods that are closely related to our work.

2.1. Deepfake Creation Methods

In this section, we explore deepfake creation methods using Generative Adversarial Networks (GANs), acknowledging that other techniques also exist for generating forgeries.

1. *Creation of forgeries with GANs*

The Generative Adversarial Network (GAN) [2] is a machine learning framework that consists of two competing neural networks: the generator (G) and the discriminator (D). These networks engage in a zero-sum game [14–16].

The generator produces outputs without learning the features of the input training dataset, meaning it does not grasp the semantics of the data. Meanwhile, the discriminator evaluates both the training samples and the generated samples, training to maximize the probability of correctly classifying real training samples as genuine and generated samples as fake.

During training, the generator continually refines its outputs to create realistic images that can deceive the discriminator, while the discriminator works to accurately distinguish between real and generated images. Both networks are optimized simultaneously using

cross-entropy loss functions in a two-player minimax game, with $\Lambda(D;G)$ serving as the value function [17].

$$\min_G \max_D \Gamma(G,D) = E_{x \in p_{data}}[log(D(x))] + E_{z \in p_{gen}}[log(1 - D(G(z)))] \quad (1)$$

It is thus inevitable to obtain better results in a model where two different deep learning networks compete against each other.

The generator is also sometimes known as the decoder, while the discriminator is alternatively known as the encoder or parser.

Various types of GANs have been developed for face and body alteration and forgery. RSGAN [18] and StarGAN [19] perform face replacement using facial and hair features, and facial features and facial expressions, respectively. GANimation [20] performs synthesis while transferring the contours of a face without distorting the transferred identity. PG-GAN [9] produces hyper-realistic fakes while BiGAN [4] produces fakes that are difficult to detect due to its use of a random noise variable received by the encoder in addition to the GAN structure. CBiGAN [21] was developed to minimize BiGAN's generalization losses and complete the learning process at lower power consumption. It has been stated that WGAN [22] provides better quality output due to improved stability in the optimization process and the correlation between the convergence of the generator and the loss function compared to the standard GAN model [14].

Figure 1 shows examples of faces generated by InterfaceGAN, ProGAN, StyleGAN3, and a real image from the CelebA dataset.

Figure 1. From left to right: Examples of faces generated by InterfaceGAN, ProGAN, StyleGAN3, and a real image from the CelebA dataset.

2. *Why are GAN artifacts produced?*

Artifacts, also interchangeably known as residues or remnants, are generated by the manipulation techniques applied within the generator neural network of a GAN. The process begins with the generator receiving a noise input, denoted as z, which varies throughout the network's operation. This noise input, which ranges from zero to one, can lead to the creation of similar synthetic images when the values are close or identical. Additionally, the upsampling methods used in many GAN models can introduce artifacts or similarities in the generated images. Notably, applying consistent modifications to the same areas in the output images enables the detection of GAN-generated content through repetition and generalization analysis [23]. It is important to note that GAN artifacts are not solely a result of model collapse—where the model produces identical or nearly identical images—or convergence failure, where the generator loss value approaches zero and results in meaningless outputs. Instead, GAN artifacts can arise from various factors, including the internal architecture of the GAN, the diversity of the training data, the size and quality of the dataset, and the resolution of the images used for training.

2.2. Deepfake Detection Methods

The detection of fake images, particularly those generated by Generative Adversarial Networks (GANs), relies on identifying synthetic signatures left in the images. While visible artifacts can often be detected by the human eye, invisible markers necessitate specialized forensic techniques. Various methods have emerged in this field, including assessments of human visual performance, GAN fingerprint analysis, spatial residuals, anomaly detection, frequency space examination, and co-occurrence matrix analysis.

A comprehensive literature review reveals that deepfake detection methods are more prevalent than creation methods, with studies indicating 11,111 detection-related publications compared to 9752 on creation [24].

Recent research indicates a significant reliance on deep learning for synthetic media detection, with studies showing that 70% of detection methods utilize deep learning, 20% employ traditional machine learning, and the remaining 10% focus on statistical learning [25]. Within deep learning approaches, convolutional neural networks (CNNs) dominate the landscape, comprising 78% of studies, followed by recurrent neural networks (RNNs) and long short-term memory networks (LSTMs) at 12%, with only 2% investigating region-based CNNs [26–28]. Machine learning methods, such as Gaussian Mixture Models (GMMs) and Support Vector Machines (SVMs), are also employed to enhance detection accuracy. In this study, we focus on detection of fake images of human faces. The creation of facial deepfakes is categorized under four main headings: whole face synthesis, identity modification, expression alteration, and attribute manipulation [29]. A database that allows for ongoing access to updated classification and detection strategies is actively maintained at "Awesome Face Forgery Generation and Detection" [30,31]. Notable GAN models involved in face synthesis include DCGAN, WGAN, PGGAN, and StyleGAN.

Detection methods specifically aimed at identifying GAN-generated faces can be categorized into four groups: deep learning-based approaches, feature-based identifications, physiological feature analysis, and assessments of human visual performance [3,32]. Research has explored the detection of deepfakes based on physiological features, such as eye color and iris shape, using images generated by StyleGAN [33,34]. Additionally, studies have shown that facial line defects can be distinguishing features between real and synthetic images, particularly in children, who tend to exhibit smoother facial transitions [35].

Earlier studies focusing on physical attributes highlighted various detection techniques, including camera parameter analysis, interpolation effects, compression artifacts, and visual inconsistencies in eye reflections [36]. These foundational studies paved the way for more recent research that validates these detection methods for StyleGAN while noting that StyleGAN2 has effectively addressed many of the inconsistencies and artifacts identified in previous models [37].

A significant challenge in deepfake detection is the variability in effectiveness across different GAN models. The generalization of detection methods can be compromised if images are generated by models that differ from those used in the training phase. This inconsistency may arise from the specific training datasets employed during the learning phase of the GAN.

Overall, deepfake detection remains a rapidly advancing area of research, driven by the increasing sophistication of generative models and the need for robust verification methods.

3. Taxonomy of GAN Artifacts

The study and classification of GAN artifacts have gained significant attention in recent years. Influential studies [38,39] laid the groundwork for this area of research, establishing a systematic approach to understanding the artifacts produced by GANs.

GAN artifacts can be broadly categorized into two types: those distinguishable by the human eye and those that are not [3]. Visible artifacts are detectable through human visual perception, whereas invisible artifacts can only be identified through digital analysis methods. Figure 2 illustrates the taxonomy of GAN artifacts.

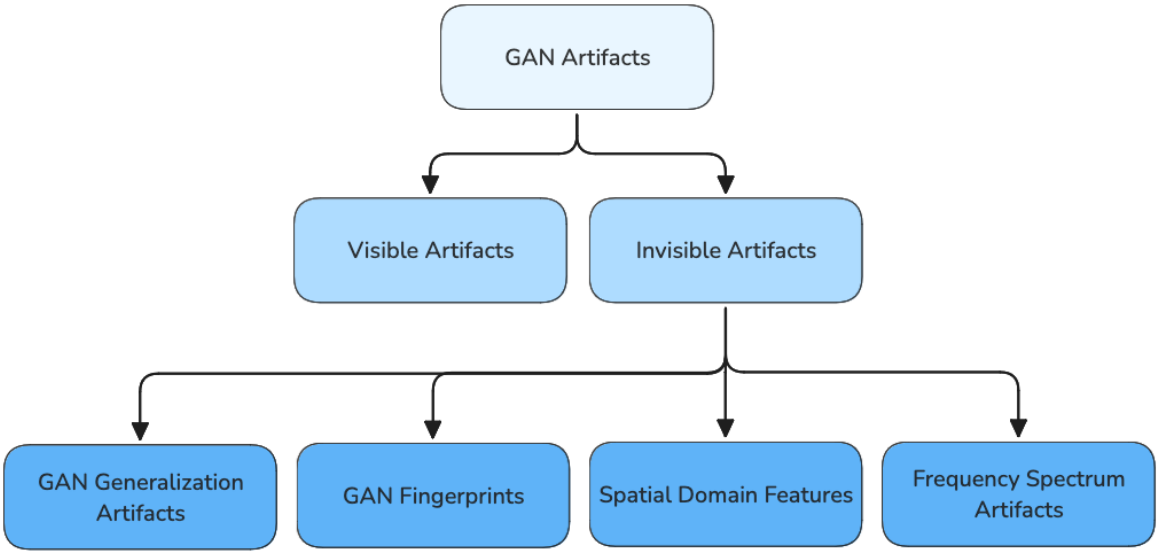

Figure 2. GAN artifacts can be broadly classified as visible and invisible. The latter can be further divided into four classes.

Invisible GAN artifacts are further divided into four classes:

1. Residues from GAN Generalization: These artifacts arise when a GAN struggles to learn the distribution of the training data, often due to convergence issues between learning and testing phases. Current GAN objectives and their internal architecture do not adequately promote a wide variety of distributions in synthetic outputs [40]. While this situation may seem advantageous because the parser effectively generalizes the learning process, it also fails to recognize the lack of diversity in outputs [41]. This can prevent the generation of diverse synthetic outputs, suggesting that simply increasing the quantity and diversity of training data may not suffice. Some GANs struggle with generating realistic content when predefined patterns, like typical room layouts, are not present. These phenomena highlight the necessity for improvements in GAN architecture to enhance generalization capabilities.
2. GAN Fingerprints: Each GAN model leaves unique fingerprints on the synthetic images it generates, detectable through model-based or image-based analysis. Characteristics such as training set distribution, network architecture, hyperparameters, and optimization functions contribute to these fingerprints. Distinctive marks are consistently left by various GANs across different architectures and datasets [42], which can be represented as an RGB image of the same dimensions as the original [43], thereby validating the concept of GAN fingerprints. The study of [44] also states that the GAN fingerprint can be uniquely detected according to the GAN model and that the obtained fingerprint depends on the dataset used during training.
3. Spatial Domain Features: Artifacts in this category stem from the inherent limitations of the GAN model. For instance, GANs may struggle to create images with proper saturation or to replicate the varied exposure levels typical of real-world cameras, leading to the absence of certain photographic attributes in their outputs [45]. Other studies [46] have focused on identifying synthetic images by examining the co-occurrence of RGB pixels. Moreover, since GANs generally operate in the RGB color

space, detection can be performed using statistical methods that analyze chrominance values from the residues in RGB, HSV, and YCbCr color spaces [47].

4. Frequency Spectrum Artifacts: This category of artifacts is identified through frequency space analysis, which transforms synthetic images to reveal distinctive patterns. Research has shown that Discrete Cosine Transform (DCT) methods can effectively differentiate between real and synthetic images, as well as identify the specific GAN model used for their generation [48,49]. Additionally, Fourier transforms have been employed for deepfake detection [50], and watermarking techniques in frequency space have been developed to safeguard intellectual property [51].

Understanding the taxonomy of GAN artifacts is crucial for improving the detection of synthetic images. Ongoing research is uncovering the complexities of these artifacts, offering insights that can enhance the technology's capabilities and its applications across various fields. In this study, we will exploit frequency spectrum artifacts, a subset of invisible artifacts, to develop feature extraction methods; after applying these methods, we will be able to use simple machine learning classifiers to detect deepfakes with great ease and accuracy.

4. Methodology of the Novel Deepfake Detection Framework

This study introduces a new approach for detecting high-quality deepfakes generated by unfamiliar (unseen or unknown) GAN models.

4.1. Proposed Framework

The goal of the new framework is to identify common features and artifacts from different GANs to improve detection capabilities, and to use the most effective methods to classify images as real or fake. Thus, the detection process consists of two important steps: Data Preparation and Classification. Our framework applies all data preparation and classification techniques to images or datasets, yielding results from all combinations. When a new image or dataset is provided, the detection pipeline processes all combinations alongside genuine images to ensure reliable results.

- **Data Preparation**: Before assessing whether a raw image is fake, it must go through preprocessing. This step includes extracting features from the image and scaling them to ensure that each feature holds equal significance in the analysis. Various methods can be used for feature extraction; in this study, we will employ a combination of Fourier transform, wavelet transform, and Histogram of Oriented Gradients (HOG) 1D transform, alongside pairwise Red, Green, and Blue (RGB) color correlations.
- **Binary Classification**: After features are usefully extracted, a classifier algorithm is employed to categorize the image either as real or fake [23].

In our experiment, we utilized several classifiers, including Gaussian Mixture Model, XGBoost, Random Neural Network, Support Vector Machines (SVM), Logistic Regression, K Nearest Neighbours, Gradient Boosting, Naive Bayes, AdaBoost, and Extra Trees. Each model contributes to the creation of a robust deepfake detection pipeline. The combination of different feature extraction methods (Fourier transform, wavelet transform, and HOG transform, along with color differences) with the advanced classifiers mentioned above ensures that both subtle and obvious manipulations introduced by deepfake generation are detected accurately. Figure 3 illustrates how the detection pipeline works.

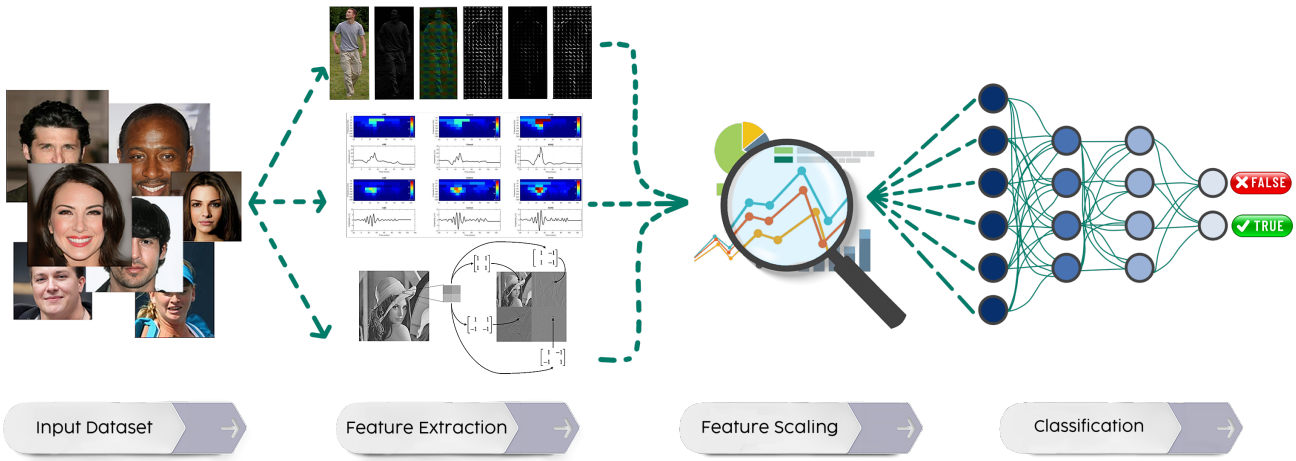

Figure 3. Detection pipeline.

4.2. Data Preparation

Data preparation consists of two important steps described below: (i) Feature Extraction, and (ii) Feature Scaling.

1. **Feature Extraction**
   Our innovative approach to extracting meaningful features from images involves combining frequency space transformations—such as Fourier, wavelet, and Histogram of Oriented Gradients (HOG)—with metrics from the Red–Green–Blue (RGB) color channels. As demonstrated in a later section, combining multiple image processing methods for feature extraction hugely enhances deepfake detection effectiveness. Figure 4 illustrates the feature extraction process.

   - **Fourier Transform for Image Processing**: The Fourier transform (FT) is a mathematical method that decomposes a signal into its individual frequencies. In image processing, the 2D Fourier transform converts an image from the spatial domain (where pixels are arranged by location) to the frequency domain, where each point represents a specific frequency and amplitude. This technique is particularly useful for deepfake detection, as the generation of deepfakes often introduces subtle frequency domain artifacts, such as compression artifacts (e.g., ringing, blurring, jagged edges, and color degradation) [52] and blending or smoothing artifacts from manipulations like face-swapping. The Fourier transform helps uncover these anomalies by highlighting unnatural patterns. The feature extraction process involves three main steps [53]: (i) Spatial Domain to Frequency Domain Conversion: The FT separates an image into frequency components, where low frequencies represent broad patterns such as skin and high frequencies capture fine details such as edges, wrinkles, and hair. (ii) Logarithmic Transformation: A logarithmic transformation (log(abs(FT))) is applied to enhance visibility, and convert the wide dynamic range generated by the FT, where high-frequency components tend to be small, and low-frequency components are dominant, into a more interpretable range for analysis. (iii) Shift to Center Frequencies: The Fourier Shift (fftshift) repositions the zero-frequency component to the center of the spectrum, facilitating easier interpretation during analysis. Figure 5 shows the principal component analysis (PCA) of images after applying the Fourier transform on each dataset.

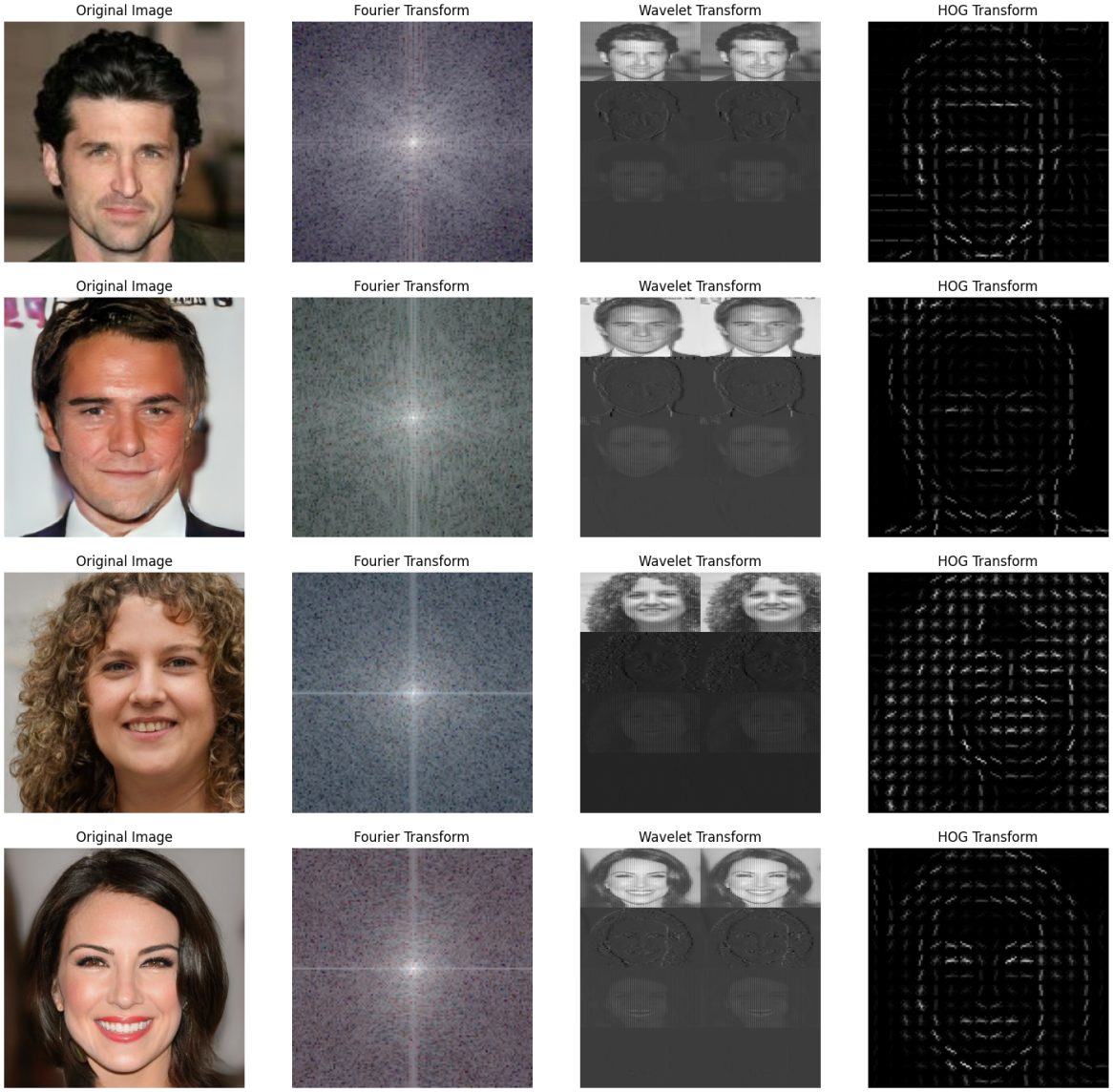

Figure 4. From the top: The Fourier, wavelet, and HOG transforms applied to one image from the CelebA, ProGAN, StyleGAN3, and InterfaceGAN datasets.

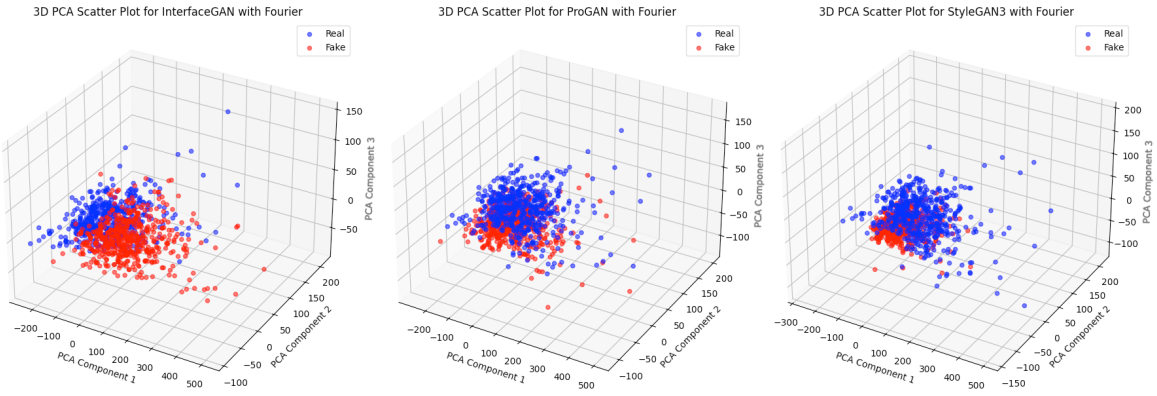

Figure 5. Principal Component Analysis of images after application of the Fourier transform on each dataset.

- **Wavelet Transform for Image Processing**: The wavelet transform is a powerful mathematical tool that decomposes images into frequency components, enabling the identification of textures, edges, and patterns. A wavelet is a compact os-

cillatory function that starts at zero, varies in amplitude, and returns to zero, making it distinct from traditional infinite functions like sine and cosine used in Fourier transforms. This finite duration allows wavelets to capture localized features, making them particularly effective for analyzing signals with varying frequencies and behaviors at different scales [23]. Wavelets can be used to analyze and represent data in both time and frequency domains. Wavelets offer unique characteristics, such as sparse wavelet sub-bands [54], which enhance image representation quality through multi-scale modeling and enable efficient representation and storage of multi-resolution images. For instance, the high-frequency sub-bands contain most of the texture information of the image. Wavelets are robust against shifts and distortions, enhancing their practical applicability [55]. However, wavelet transforms can be computationally intensive, and the choice of wavelet function and parameters requires careful consideration, as these factors can significantly influence the results. Figure 6 shows the principal component analysis (PCA) of images after applying the wavelet transform on each dataset.

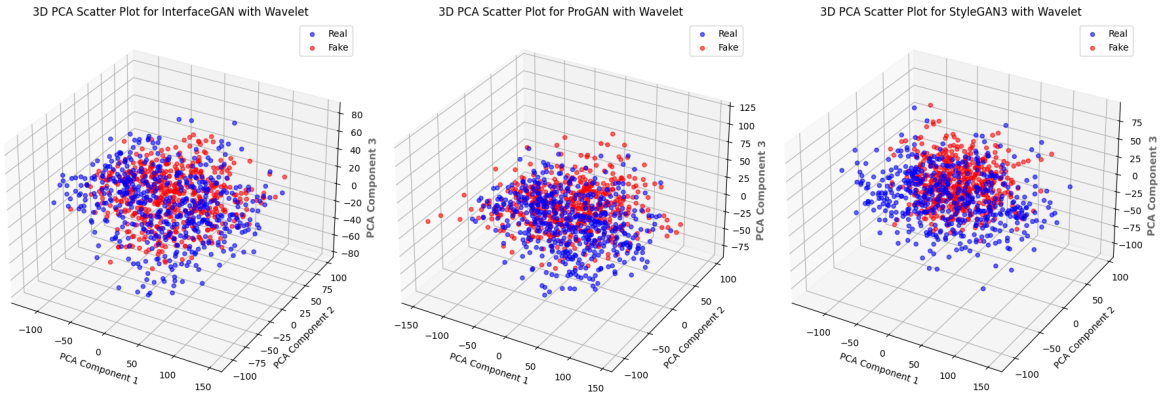

Figure 6. Principal Component Analysis of images after application of the wavelet transform on each dataset.

- **HOG Transform for Image Processing**: The Histogram of Oriented Gradients (HOG) is a feature extraction technique used in image analysis to capture object shape and structure by analyzing edge orientations [56]. It was first used in face detection by Dalal and Triggs [57]. HOG divides the image into small regions (cells), computes gradient orientations, and creates histograms that summarize edge directions. These histograms are combined into feature vectors for object detection and recognition, making HOG effective at handling variations in lighting and contrast. For example, in the case of a blueberry, the consistent gradient orientations in its round shape make classification easier [58]. HOG is similar to techniques like edge orientation histograms, scale-invariant feature transform (SIFT), and shape contexts but differs by using a dense grid and local contrast normalization for improved accuracy. However, it has limitations, such as sensitivity to image transformations, reliance on local gradient data, and a fixed grid structure that may reduce its discriminative power in some cases. Figure 7 shows the principal component analysis (PCA) of images after applying the HOG transform on each dataset.

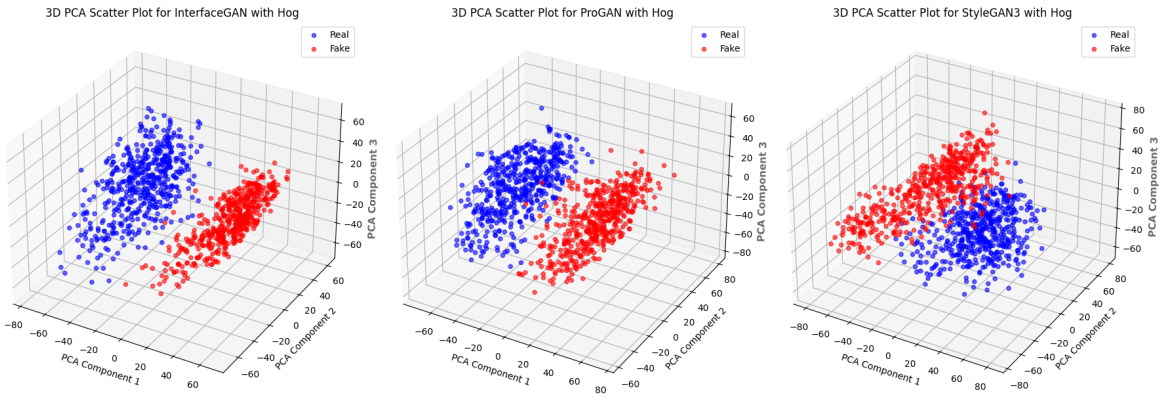

Figure 7. Principal Component Analysis of images after application of the HOG transform on each dataset.

- **Color Difference Computation**: The human face has a well-defined color structure, such as in skin tones. Deepfake images often disrupt the natural relationships among color channels, especially following techniques like smoothing or blending. Analyzing these color differences can therefore reveal signs of tampering. The color difference function works on the Red, Green, and Blue (RGB) channels of an image after applying Fourier, wavelet, or HOG transforms to compute the following statistical features that describe the relationships between the color channels in the frequency domain.
  - *Mean, Min, and Max Differences*: Mean difference is the average magnitude of the difference between color channels across all pixels. Min/Max differences are the smallest or largest color channel differences, which may indicate anomalies in blending or compression.
  - *Correlation Between Channels*: In natural images, the RGB channels exhibit a certain level of correlation due to how light and color interact on real-world surfaces such as human skin. When a deepfake is generated, these correlations can be altered, especially if facial features are sourced from different images. The color difference function computes the pairwise correlation coefficients between the color channels. The formula 1 - corr(R, G) (or similar for the other channels) is used to capture the degree of uncorrelated behavior between the channels. If the channels are highly correlated, this value will be close to zero. A higher value indicates a stronger divergence between the channels, which might suggest image tampering.
  - *Frequency Domain Anomalies*: Unnatural correlations occur when one color channel is manipulated independently of the others. Artificial enhancements or color shifts are introduced when artificial color adjustments, such as changing skin tone to match lighting, are made in the images.

2. **Feature Scaling**

Feature scaling is an essential preprocessing step in machine learning that ensures each feature contributes equally to model performance by bringing them into a similar range. In deepfake detection, where datasets include diverse features from raw pixel values to frequency domain values and color metrics, proper scaling is vital. For example, Fourier-transformed features may have values spanning several orders of magnitude, while color difference metrics might vary within a much smaller range. Similarly, a dataset may include pixel intensity (0 to 255) and age (0 to 100). Without scaling, certain features may carry disproportionate weight, resulting in inaccurate classification. Scaling ensures all features are treated equally, allowing the model to capture the true variance of the dataset without

bias. It also optimizes performance, enabling algorithms like the Gaussian Mixture Model to converge more effectively and accurately model probability distributions, ultimately improving classification of real vs. deepfake images.

The two common types of feature scaling are the following:

- Min–Max Scaling (Normalization) is defined by the formula $x' = \frac{x - \min(x)}{\max(x) - \min(x)}$. This method scales data to a range between 0 and 1, making it useful when the data distribution is non-Gaussian or when algorithms need bounded inputs, such as neural networks. It is ideal for algorithms that require inputs within a specific range, like [0, 1] or [−1, 1].
- Z-Score Normalization (Standardization) follows the formula $z = \frac{x - \mu}{\sigma}$, where μ is the mean and σ is the standard deviation. This technique transforms data to have a mean of 0 and a standard deviation of 1, which is particularly suitable for features that follow a Gaussian distribution. It is commonly used in algorithms such as Gaussian Mixture Models (GMMs), Support Vector Machines (SVMs), and Principal Component Analysis (PCA), all of which are sensitive to the magnitude of the data.

4.3. Classification Techniques

Once the image data are preprocessed and features are extracted and scaled, the images are ready for classification. Classifiers vary in complexity and assumptions about the data, affecting their suitability for different tasks.

Simple models like Logistic Regression and k-Nearest Neighbors (k-NNs) are easy to interpret but have limitations, such as assuming linear relationships and being computationally expensive with larger datasets. k-NNs is also sensitive to feature scaling. Naive Bayes is fast and effective but assumes feature independence, a condition that may not always hold true in practice. Random Forests and Extra Trees are valuable for understanding feature importance, especially with controlled tree depth. Support Vector Machines (SVMs) excel at handling high-dimensional data, while Gaussian Mixture Models are great for managing complex data distributions with clustering and overlaps. Neural networks are powerful but computationally demanding and less interpretable. Lastly, XGBoost, Gradient Boosting, and AdaBoost offer high accuracy but require careful tuning. Below is a brief description of the classifiers used in our experiments.

- The **Gaussian Mixture Model (GMM)** is a probabilistic model that classifies data as a mixture of Gaussian distributions. It treats real and fake images as mixtures based on features like color channel differences and frequency domain patterns. GMM performs soft classification, estimating the likelihood that an image belongs to a class. This helps to handle uncertainty and overlap between the two classes, thus being especially good for detecting realistic fakes. It is especially effective for identifying frequency domain anomalies. Training uses the Expectation–Maximization (EM) algorithm to optimize the model's parameters to maximize the likelihood of the observed data [50].
- **Gradient Boosting, XGBoost, and ADA Boost** are ensemble methods that combine weak learners, usually decision trees, to improve predictions. Gradient Boosting builds trees sequentially, correcting errors using the gradient of the loss function. AdaBoost focuses on misclassified instances, giving them higher weights. XGBoost enhances Gradient Boosting with regularization to reduce overfitting and improve speed, handling sparse data and missing values effectively. It excels with structured data and complex non-linear relationships.
- **Random Forest** is an ensemble method that builds multiple decision trees using random subsets of data and features, enhancing robustness. By adjusting tree depth, it prevents overfitting, focusing on key features. It also offers interpretability through feature importance metrics, helping identify influential features. Random Forest handles high-

dimensional data, such as large feature sets from frequency transformations and color-difference analyses, effectively [59].

- **Neural Networks (NNs)** with multiple dense layers automatically learn relevant features from frequency domain and color-difference metrics [34]. They capture both global patterns (e.g., face shape) and subtle local differences (e.g., blending artifacts). Neural Networks are scalable, improving performance with larger datasets and more complex features, but they lack interpretability. Our Neural Network classifier uses one input layer, two dense hidden layers with ReLU activation, and a final output layer with sigmoid activation. The binary cross-entropy loss function and cross-validation were applied.
- **Support Vector Machine (SVM)** is a powerful supervised learning model for classification and regression. It finds the optimal hyperplane that maximizes the margin between support vectors. SVM handles non-linear relationships using kernel functions, which map input data to higher dimensions for separating non-linearly separable classes. Common kernels include linear, polynomial, and RBF. SVM is particularly effective in high-dimensional spaces [59].
- **K-Nearest Neighbors (KNNs)** is a simple, non-parametric algorithm for classification and regression. It predicts based on the 'k' closest training points, using majority vote (classification) or averaging (regression). KNNs is easy to implement and adapts well to the structure of the data but is sensitive to the choice of 'k' and irrelevant features. It can also be slow for large datasets, as it calculates distances to all points during prediction [59].
- **Logistic Regression** predicts the probability of an input belonging to a category by applying the logistic function to a linear combination of features, yielding a value between 0 and 1. It is simple, interpretable, and efficient for small datasets but sensitive to outliers and assumes linearity, limiting its ability to model complex, non-linear relationships. It can also struggle with high-dimensional or highly correlated data, risking overfitting. For face deepfake detection, it has been used in [34,60] as a classifier.
- **Naive Bayes** uses Bayes' theorem to calculate class probabilities, assuming feature independence. This makes it efficient in speed and memory, ideal for large datasets. However, it struggles with correlated features and zero probabilities, often requiring techniques like Laplace smoothing for unseen feature-class combinations.
- **Extra Trees (Extremely Randomized Trees)** is an ensemble method that builds decision trees with added randomness, selecting features and split thresholds randomly. This reduces overfitting and improves generalization. Extra Trees are faster to train than methods like Random Forests while maintaining high accuracy, making them effective for large datasets and providing useful feature importance insights [59].

5. Experimental Implementation of Multi-GAN Deepfake Detection Framework

5.1. Test Environment

The simulation was designed to run in the Google Colab (Colaboratory) Python 3 environment with High RAM (51 GB). The use of Jupyter Notebooks and GPU was found appropriate for ease of operation and speed. Colab Pro+ was used in the creation of GAN models of the datasets to be presented for the simulation, training of the simulation, and analysis stages. Colab has been used in various deep learning studies and has facilitated studies in this field thanks to GPU, TPU, Python runtime, RAM, and disk [61–64]. In addition to the features it provides to the working environment, the datasets used in this field are readily available in the Colab environment; it is possible to use these datasets and import them with one line of code.

5.2. Base Datasets and GAN Models

CelebA [65] dataset was used for real faces. Since there are no directly accessible deepfake facial image datasets for ProGAN, StyleGAN3, and InterFaceGAN, we created datasets of fake facial images for each of these models with an image resolution of 256 pixels by 256 pixels in JPEG format. We refer to this as the multi-GAN dataset. In creating our novel dataset, CelebA served two purposes. (i) CelebA images were used as benchmarks to test our framework. In particular, a total of 5000 randomly selected images from CelebA were used for benchmarking. (ii), CelebA images were used to help train some of the GAN models before producing novel fake facial images. In particular, for ProGAN, we used a pre-trained model from Facebook Research that was previously trained on the complete CelebA dataset: 'facebookresearch/pytorch_GAN_zoo:hub', 'PGAN', model_name = 'celebAHQ-512'.

To generate fakes by InterfaceGAN, transfer learning was adopted by first using a pretrained model of the StyleGAN CelebA HQQ faces dataset with ATTRS = ['age', 'eyeglasses', 'gender', 'pose', 'smile']. It was deemed appropriate to use various accessories such as glasses and sunglasses in photographs of faces of different ages, genders, ethnic groups and backgrounds, and different poses and smiles. This dataset was not directly presented to the simulation, but was presented to the simulation as a dataset for detection to indicate real images during the comparison process.

The pretrained model used to generate the fake images of the StyleGAN3 model was stylegan3-t-ffhq-1024x1024.pkl.

More details about the datasets used are listed in Table 1.

Table 1. Datasets used in this work include 5000 images each from CelebA and 3 newly generated fake image datasets using StyleGAN3, ProGAN, and InterfaceGAN.

Authenticity	Model	Piece	Source	Time to Generate	Size	Type and Preprocessed Resolution
Real	CelebA	5000	CelebA veri seti	Randomly picked	35 MB	JPEG, 256 × 256
Fake	StyleGAN3	5000	Produced by us	15 min	119 MB	JPEG, 256 × 256
Fake	InterfaceGAN	5000	Produced by us	35 min	1.1 GB	JPEG, 1024 × 1024
Fake	ProGAN	5000	Produced by us	6 min	123 MB	JPEG, 256 × 256

6. Results

The results of our experiment will be presented in the following sequence. In Section 6.1, we begin by detailing the metrics used to assess the performance of various classifiers and transformations across the three datasets. Next, we provide tables showing the accuracy and the time taken to achieve those results for each dataset, with the time measurements reflecting the evaluation of all metrics described in Section 6.1. Following this, we present data on the remaining metrics for each GAN model. This approach is used because the primary trends in the results can be readily observed from the accuracy vs. time data, and the other metrics generally reflect similar patterns. This separation enhances the clarity of the visual data presentation.

6.1. Metrics to Evaluate Model Performance

The performance of the fake image detection pipeline is evaluated across the three transforms and various classifiers using the following metrics [66]:

- Accuracy is the fraction of correct predictions among all predictions. The formula for accuracy is $\frac{TP+TN}{TP+TN+FP+FN}$, where TP, TN, FP, and FN are true positive, true negative, false positive, and false negative, respectively.
- Precision measures the proportion of true positive predictions among all instances that were predicted as positive by the model. It answers the following question: "Of all the instances the classifier predicted as positive, how many were actually positive?" The formula for precision is $\frac{TP}{TP+FP}$.
- Recall (also known as sensitivity, true positive rate, or hit rate) is a performance metric that evaluates the ability of a classifier to correctly identify all relevant instances of a particular class. Specifically, it measures the proportion of actual positive instances that were correctly predicted by the classifier. In other words, recall answers the following question: "Of all the actual positive instances, how many did the model correctly identify?" The formula for recall is $\frac{TP}{TP+FN}$.
- The F1 score is the harmonic mean of precision and recall: $F1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision}+\text{Recall}}$

6.2. Accuracy Versus Time

In this section, we showcase our results in terms of accuracy and time taken to generate those results. Table 2, Table 3 and Table 4 shows the accuracy and time taken for each classifier and transform method against our datasets StyleGAN3, ProGAN and InterfaceGAN respectively.

Table 2. StyleGAN3.

Transform		**Fourier**		**Wavelet**		**HOG**
Classifier	**Accuracy**	**Time (in sec)**	**Accuracy**	**Time (in sec)**	**Accuracy**	**Time (in sec)**
Naive Bayes	0.8884	0.03	0.8715	0.03	0.9535	0.03
Log. Regression	0.9103	0.08	0.9666	0.11	0.9892	0.09
KNN	0.9073	0.74	0.9636	0.75	0.9986	0.76
Grad Boost	0.9562	33.27	0.9902	33.48	0.9992	33.38
AdaBoost	0.9212	10.24	0.9644	12.52	0.9965	10.19
XGBoost	0.9525	0.36	0.9887	1.79	0.9982	0.33
Random Forest	0.908	5.56	0.9627	5.47	0.9868	5.49
Extra Trees	0.9249	0.65	0.9593	0.73	0.9869	0.66
SVM	0.9633	3.39	0.9958	2.14	0.9983	0.97
SVM with RBF Kernel	1	435.39	1	444.22	1	441.12
NeuralNetwork	0.9954	7.38	1	7.42	1	7.69
GMM	0.1773	0.85	0.3066	2.49	0.6711	1.99

Table 3. ProGAN.

Transform	Fourier		Wavelet		HOG	
Classifier	**Accuracy**	**Time (in sec)**	**Accuracy**	**Time (in sec)**	**Accuracy**	**Time (in sec)**
Naive Bayes	0.8817	0.03	0.909	0.04	0.969	0.04
Log. Regression	0.8975	0.1	0.9885	0.12	0.9991	0.07
KNN	0.9131	0.88	0.9716	0.76	0.9979	0.74
Grad Boost	0.9483	33.43	0.996	33.25	0.9998	33.6
AdaBoost	0.9021	10.13	0.9827	10.18	1	10.21
XGBoost	0.9453	1.96	0.9931	0.34	0.9999	0.33
Random Forest	0.882	5.43	0.974	5.38	0.9841	5.51
Extra Trees	0.9066	0.66	0.9803	0.66	0.9942	0.67
SVM	0.9525	3.95	0.9975	1.89	0.9997	0.74
SVM with RBF Kernel	1	439.38	1	441.86	1	439.51
NeuralNetwork	0.9929	6.8	0.9989	8.04	1	7.48
GMM	0.181	1.14	0.4806	2.45	0.997	0.74

Table 4. InterfaceGAN.

Transform	Fourier		Wavelet		HOG	
Classifier	**Accuracy**	**Time (in sec)**	**Accuracy**	**Time (in sec)**	**Accuracy**	**Time (in sec)**
Naive Bayes	0.9211	0.03	0.9578	0.03	0.9888	0.03
Log. Regression	0.9847	0.11	1	0.08	1	0.06
KNN	0.9825	0.87	0.9903	0.76	0.9986	0.78
Grad Boost	0.9947	33.45	0.9996	32.95	1	33.55
AdaBoost	0.9894	10.19	0.9979	10.23	1	10.09
XGBoost	0.9954	1.75	0.9991	0.38	1	1.54
Random Forest	0.969	5.54	0.9845	5.4	0.9948	5.45
Extra Trees	0.9765	0.63	0.9839	0.64	0.9963	0.62
SVM	0.9916	1.21	0.9997	1.3	0.9999	0.54
SVM with RBF Kernel	1	670.29	1	681.27	1	660.09
NeuralNetwork	1	7.8	1	7.32	1	7.55
GMM	0.9741	0.72	0.5055	1.97	0.9978	0.63

6.3. Recall, Precision, F1 Score

In this section, we report the recall, precision, and F1 score for each classifier. Table 5, Table 6, and Table 7 report the recall, precision, and F1 score for StyleGAN3, ProGAN, and InterfaceGAN, respectively.

Table 5. StyleGAN3 results for recall, precision, and F1 score.

Transform		**Fourier**			**Wavelet**			**HOG**	
Classifier	**Recall**	**Precision**	**F1 Score**	**Recall**	**Precision**	**F1 Score**	**Recall**	**Precision**	**F1 Score**
Naive Bayes	0.8884	0.8884	0.8884	0.8715	0.8731	0.8714	0.9535	0.9538	0.9535
Log. Regression	0.9103	0.9105	0.9103	0.9666	0.9666	0.9666	0.9892	0.9892	0.9892
KNN	0.9073	0.9165	0.9068	0.9636	0.9646	0.9636	0.9986	0.9986	0.9986
Grad Boost	0.9562	0.9566	0.9562	0.9902	0.9902	0.9902	0.9992	0.9992	0.9992
AdaBoost	0.9212	0.9216	0.9212	0.9644	0.9644	0.9644	0.9965	0.9965	0.9965
XGBoost	0.9525	0.9534	0.9525	0.9887	0.9887	0.9887	0.9982	0.9982	0.9982
Random Forest	0.908	0.911	0.9078	0.9627	0.9628	0.9627	0.9868	0.9868	0.9868
Extra Trees	0.9249	0.9283	0.9248	0.9593	0.9599	0.9593	0.9869	0.9869	0.9869
SVM	0.9633	0.9636	0.9633	0.9958	0.9958	0.9958	0.9983	0.9983	0.9983
SVM with RBF Kernel	1	1	1	1	1	1	1	1	1
Neural Network	0.9954	0.9954	0.9954	1	1	1	1	1	1
GMM	0.1773	0.1318	0.1511	0.3066	0.2188	0.2479	0.6711	0.801	0.6313

Table 6. ProGAN results for recall, precision, and F1 score.

Transform		**Fourier**			**Wavelet**			**HOG**	
Classifier	**Recall**	**Precision**	**F1 Score**	**Recall**	**Precision**	**F1 Score**	**Recall**	**Precision**	**F1 Score**
Naive Bayes	0.8817	0.8825	0.8816	0.909	0.909	0.909	0.969	0.9695	0.969
Log. Regression	0.8975	0.8975	0.8975	0.9885	0.9885	0.9885	0.9991	0.9991	0.9991
KNN	0.9131	0.9143	0.913	0.9716	0.9716	0.9716	0.9979	0.9979	0.9979
Grad Boost	0.9483	0.9489	0.9483	0.996	0.996	0.996	0.9998	0.9998	0.9998
AdaBoost	0.9021	0.9023	0.9021	0.9827	0.9827	0.9827	1	1	1
XGBoost	0.9453	0.946	0.9453	0.9931	0.9931	0.9931	0.9999	0.9999	0.9999
Random Forest	0.882	0.8875	0.8816	0.974	0.974	0.974	0.9841	0.9843	0.9841
Extra Trees	0.9066	0.9083	0.9065	0.9803	0.9803	0.9803	0.9942	0.9942	0.9942
SVM	0.9525	0.9527	0.9525	0.9975	0.9975	0.9975	0.9997	0.9997	0.9997
SVM with RBF Kernel	1	1	1	1	1	1	1	1	1
Neural Network	0.9929	0.9929	0.9929	0.9989	0.9989	0.9989	1	1	1
GMM	0.181	0.1372	0.1555	0.4806	0.4805	0.4799	0.997	0.997	0.997

Table 7. InterfaceGAN results for recall, precision, and F1 score.

Transform	Fourier			Wavelet			HOG		
Classifier	**Recall**	**Precision**	**F1 Score**	**Recall**	**Precision**	**F1 Score**	**Recall**	**Precision**	**F1 Score**
Naive Bayes	0.9211	0.9246	0.9209	0.9578	0.959	0.9578	0.9888	0.9889	0.9888
Log. Regression	0.9847	0.9847	0.9847	1	1	1	1	1	1
KNN	0.9825	0.9826	0.9825	0.9903	0.9903	0.9903	0.9986	0.9986	0.9986
Grad Boost	0.9947	0.9947	0.9947	0.9996	0.9996	0.9996	1	1	1
AdaBoost	0.9894	0.9894	0.9894	0.9979	0.9979	0.9979	1	1	1
XGBoost	0.9954	0.9954	0.9954	0.9991	0.9991	0.9991	1	1	1
Random Forest	0.969	0.9693	0.969	0.9845	0.9848	0.9845	0.9948	0.9948	0.9948
Extra Trees	0.9765	0.9766	0.9765	0.9839	0.9842	0.9839	0.9963	0.9963	0.9963
SVM	0.9916	0.9916	0.9916	0.9997	0.9997	0.9997	0.9999	0.9999	0.9999
SVM with RBF Kernel	1	1	1	1	1	1	1	1	1
Neural Network	1	1	1	1	1	1	1	1	1
GMM	0.9741	0.9748	0.9741	0.5055	0.5055	0.5047	0.9978	0.9978	0.9978

7. Discussion

7.1. Algorithmic Efficiency and Speed Performance Comparisons

- As seen from Tables 2–4, Naive Bayes is the fastest algorithm, taking only approximately 0.03 s. It outperforms all other classifiers by at least an order of magnitude while still delivering excellent average accuracies of 0.89, 0.91, and 0.97 for the StyleGAN3, ProGAN, and InterfaceGAN datasets, respectively.
- Logistic Regression is two to three times slower than Naive Bayes; however, it yields better results, particularly when applied to wavelet transforms.
- K-Nearest Neighbors (KNNs) is not a good choice of classifier for our datasets and algorithmic pipeline. Despite taking seven to ten times longer to compute than Logistic Regression, it generally performs a little worse. Only in the case of the ProGAN–Fourier transform does it perform marginally better.
- For a boost in accuracy compared to the classifiers discussed so far, one could use Gradient Boosting, provided one is willing to wait nearly half a minute on average. For comparison, Gradient Boosting is about one thousand times slower than the Naive Bayes classifier, and offers anywhere from seven to twelve percentage points increase in accuracy over the latter, bearing in mind that Naive Bayes already has at least 87% accuracy. AdaBoost takes only one third the time that Gradient Boosting takes but shows both better and worse dataset- and transform-dependent performance. The more efficient alternative to Gradient Boosting for similar accuracies is XGBoost, which is thirty to one hundred times faster.
- Random Forest takes over five seconds on average to execute but underperforms compared to Logistic Regression. Extra Trees takes only one-tenth the time that Random Forest takes, but performs nearly on par or better. However, it still cannot beat Logistic Regression in efficiency for nearly comparable results.
- Support Vector Machine (SVM) is ten to fifty times faster than Gradient Boosting and demonstrates better performance for Fourier and wavelet transformations in

the StyleGAN3 and ProGAN datasets. It is only marginally worse for the other combinations of dataset and transforms. But SVM cannot always outperform more efficient boosting algorithms such as XGBoost.

- SVM with RBF kernel, Gradient Boosting, and AdaBoost are the three slowest classifiers. SVM with RBF kernel takes seven minutes to classify StyleGAN3 and ProGAN datasets, and eleven minutes for InterfaceGAN.
- Neural Networks and fine-tuned SVMs consistently achieve close to hundred percent accuracy. However, fine-tuning SVMs with RBF kernal takes an extraordinarily long time—several orders of magnitude longer than any other method—while neural networks only take six to eight seconds. This is because the program is trying to optimize for the best SVM parameters. In our experiments, the best SVM parameters were found to be ('C': 10, 'gamma': 0.001) across all datasets and transforms.
- ADA Boost and Extra Trees yield nearly the same results across all datasets, but ADA Boost takes over ten seconds to run, while Extra Trees completes in less than one second.

7.2. Model Suitability by Dataset

- As seen in Figure 8, Gaussian Mixture Model (GMM) performs poorly on the StyleGAN3 dataset with all transforms. For the ProGAN dataset, GMM works only with HOG transform. For InterfaceGAN, GMM performs well with both HOG and Fourier transforms.
- Gradient Boosting performs on par or slightly better for HOG transformations in the StyleGAN3 and ProGAN datasets, as well as for all transformations with InterfaceGAN.
- XGBoost and SVM have similar accuracies. For StyleGAN3 and ProGAN, XGBoost is faster. For InterfaceGAN, for Fourier and HOG transforms, SVM is faster.

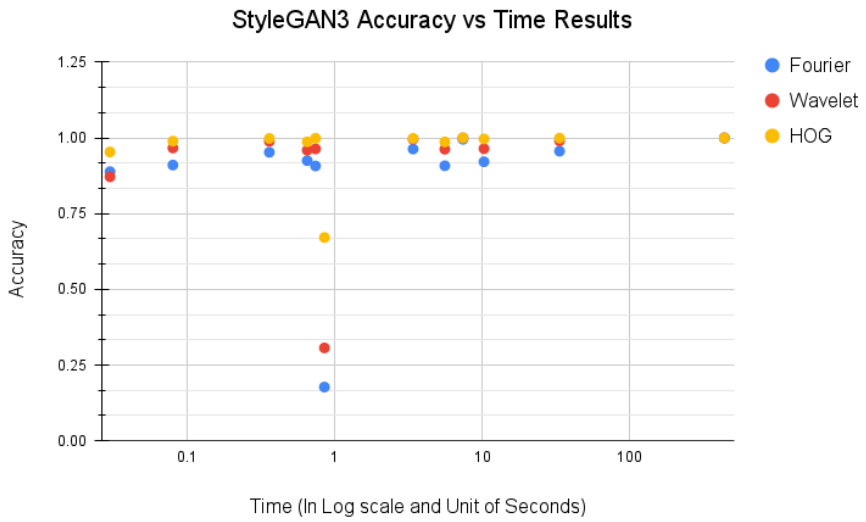

Figure 8. Accuracy vs. time for StyleGAN3 deepfake face image detection for various classifiers.

7.3. Statistics, Benchmarks, and Comparisons

Human perception studies found that participants could only discriminate fake from real images about 50% of the time [67,68]. In deep learning approaches, notable successes include an 80% detection rate using VGG-Face [69] and a 99.4% rate using CNNs for deepfake identification [70]. Further, 80% detection success was achieved for the detection of fake images produced by ProGAN and 90% and above for CycleGAN and StarGAN [71]. The use of a Deep Convolution GAN improves detection accuracy, demonstrating that even with limited datasets, effective results can be achieved through diverse data inputs [32].

In our study, we found a marked improvement in detection ability across datasets from different GAN models, across various frequency transformations as well as across various classifiers [72]. The closest comparison we can make to our work is with [23]. In comparison, our study is wider in scope in methods and metrics measured and uses different datasets. We used Fourier, wavelet, and HOG transforms in the frequency domain while they used the wavelet transform. We also used self-created datasets for StyleGAN3, ProGAN and InterfaceGAN. We measured accuracy, recall, precision and F1 score, as well as computation time, while they only reported accuracy. While [23] reported impressive detection accuracy percentages in the eighties and nineties, for all datasets, we were able to achieve 100% accuracy in at least one combination of transform and classifier.

The RealForensics detector [11], which has state-of-the-art generalization performance on unseen datasets, achieves a high accuracy of 83% and 75% for real and fake images. In comparison, our proposed method uniformly outperforms this for all the tested GANs.

8. Conclusions

The problem of detecting fake images is becoming increasingly harder due to the increased sophistication of the methods used to produce them, especially using Generative Adversarial Networks (GANs). In this study, we have made three important contributions to tackle this challenge. Covering three different transform methods, three GAN models, and twelve classification methods, ours is the most comprehensive study of detection of static deepfake face images produced by GANs.

First, we have identified and classified the various types of GAN artifacts that can help us successfully differentiate fake images from real ones. A comprehensive taxonomy categorizes these residues into visible and invisible types, with invisible residues further divided into four categories: GAN generalization artifacts, GAN fingerprints, spatial domain features, and frequency space features. In this study, we specifically explored frequency spectrum artifacts, a subset of invisible artifacts that can only be detected through digital analysis methods. We have developed feature extraction methods; after applying these, we were able to use simple machine learning classifiers to detect deepfakes with great accuracy extremely fast. Second, we have developed a mixed dataset of fake facial images, incorporating StyleGAN3, ProGAN, and InterfaceGAN, with over 15,000 images. This specialized dataset for GAN detection will support research on mixed datasets generated by different GAN models, addressing a significant gap in the current field. Third, we have developed a novel deepfake detection pipeline that analyzes digital residues in images through a combination of RGB color analysis and frequency domain transformations such as Fourier, wavelet, or HOG transforms. Using these novel feature extraction methods we achieved impressive fake image detection performance across various classification models. When evaluating newly supplied images or datasets, applying the proposed method, which encompasses all data preparation methods and classification techniques, consistently yields results. To ensure meaningful comparisons and accurate benchmarking, incorporating authentic real images is essential to validate the integrity of the analysis. Our results demonstrate significant promise in deepfake face image detection, with outstanding performance in accuracy, recall, precision, and F1 scores across various classifier algorithms, even achieving 100% accuracy in some cases. Naive Bayes, the fastest algorithm at 0.03 s, achieves at least 88% accuracy. Among the datasets, we find that InterfaceGAN fakes are easier to detect than those generated by ProGAN and StyleGAN3. When using HOG and wavelet transforms, StyleGAN3 fakes are the most challenging to detect. The Gaussian Mixture Model, however, proves to be less effective as a classifier, achieving under 20% accuracy on the StyleGAN3 dataset. Among gradient boosting techniques, XGBoost delivers the best and most time-efficient results compared to Gradient Boost and AdaBoost.

Neural networks consistently perform excellently across datasets and transforms, though they lack interpretability. The encouraging results of our work highlight several promising avenues for future research. While we have tested our detection pipeline with three GAN models, our architecture can be further tested with a variety of other models and datasets. Additionally, other image preprocessing techniques, either individually or in combination, could be investigated to improve the efficiency of classification results. Another key area for exploration is the explainability of the features extracted in our detection pipeline, which could provide deeper insights into how fake images are identified. Other state-of-the-art detection methods presented in the taxonomy could also be integrated with our framework to further enhance the accuracy of fake image detection.

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

Funding: This research received no external funding.

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors on request.

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

References

1. Xia, X.; Pan, X.; Li, N.; He, X.; Ma, L.; Zhang, X.; Ding, N. GAN-based anomaly detection: A review. *Neurocomputing* **2022**, *493*, 497–535. [CrossRef]
2. Goodfellow, I.J.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.C.; Bengio, Y. Generative adversarial networks. *Commun. ACM* **2020**, *63*, 139–144. [CrossRef]
3. Wang, X.; Guo, H.; Hu, S.; Chang, M.C.; Lyu, S. Gan-generated faces detection: A survey and new perspectives. *arXiv* **2022**, arXiv:2202.07145.
4. Donahue, J.; Krähenbühl, P.; Darrell, T. Adversarial feature learning. *arXiv* **2016**, arXiv:1605.09782.
5. Dong, C.; Kumar, A.; Liu, E. Think twice before detecting gan-generated fake images from their spectral domain imprints. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022; pp. 7865–7874.
6. Wesselkamp, V.; Rieck, K.; Arp, D.; Quiring, E. Misleading deep-fake detection with GAN fingerprints. In Proceedings of the 2022 IEEE Security and Privacy Workshops (SPW), San Francisco, CA, USA, 22–26 May 2022; pp. 59–65.
7. Huang, Y.; Xu, J.F.; Wang, R.; Guo, Q.; Ma, L.; Xie, X.; Pu, G. FakePolisher: Making deepfakes more detection-evasive by shallow reconstruction. In Proceedings of the 28th ACM International Conference on Multimedia, Seattle, WA, USA, 12–16 October 2020; pp. 1217–1226.
8. Karras, T.; Aittala, M.; Laine, S.; Härkönen, E.; Hellsten, J.; Lehtinen, J.; Aila, T. Alias-Free Generative Adversarial Networks. In Proceedings of the Advances in Neural Information Processing Systems 34 (NeurIPS 2021), Virtual, 6–14 December 2021; pp. 852–863.
9. Karras, T.; Aila, T.; Laine, S.; Lehtinen, J. Progressive Growing of GANs for Improved Quality, Stability, and Variation. In ICLR 2018, Vancouver, BC, Canada, 30 April–3 May 2018.
10. Shen, Y.; Yang, C.; Tang, X.; Zhou, B. InterFaceGAN: Interpreting the Disentangled Face Representation Learned by GANs. *IEEE Trans. Pattern Anal. Mach. Intell.* **2022**, *44*, 2004–2018. [CrossRef] [PubMed]
11. Beckmann, A.; Hillsmann, A.; Eisert, P. Fooling State-of-the-Art Deepfake Detection with High-Quality Deepfakes. In Proceedings of the ACM Workshop on Information Hiding and Multimedia Security. *arXiv* **2023**, arXiv:2305.05282v2.
12. Wang, G.; Jiang, Q.; Jin, X.; Cui, X. FFR_FD: Effective and fast detection of DeepFakes via feature point defects. *Inf. Sci.* **2022**, *596*, 472–488. [CrossRef]
13. Yan, Z.; Zhang, Y.; Fan, Y.; Wu, B. UCF: Uncovering Common Features for Generalizable Deepfake Detection. *arXiv* **2023**, arXiv:2304.13949.
14. Chaudhari, P.; Agrawal, H.; Kotecha, K. Data augmentation using MG-GAN for improved cancer classification on gene expression data. *Soft Comput.* **2020**, *24*, 11381–11391. [CrossRef]

15. Moghaddam, M.M.; Boroomand, B.; Jalali, M.; Zareian, A.; Daeijavad, A.; Manshaei, M.H.; Krunz, M. Games of GANs: Game-theoretical models for generative adversarial networks. *Artif. Intell. Rev.* **2023**, *56*, 9771–9807. [CrossRef]
16. Swathi, P.; Saritha, S.K. Deepfake creation and detection: A survey. In Proceedings of the 2021 Third International Conference on Inventive Research in Computing Applications (ICIRCA), Coimbatore, India, 2–4 September 2021; pp. 584–588.
17. Sultana, M.; Mahmood, A.; Javed, S.; Jung, S.K. Unsupervised RGBD Video Object Segmentation Using GANs. In Proceedings of the Asian Conference on Computer Vision, Perth, Australia, 2–6 December 2018.
18. Natsume, R.; Yatagawa, T.; Morishima, S. RSGAN: Face swapping and editing using face and hair representation in latent spaces. *arXiv* **2018**, arXiv:1804.03447.
19. Choi, Y.; Choi, M.; Kim, M.; Ha, J.W.; Kim, S.; Choo, J. StarGAN: Unified generative adversarial networks for multi-domain image-to-image translation. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018; pp. 8789–8797.
20. Pumarola, A.; Agudo, A.; Martinez, A.M.; Sanfeliu, A.; Moreno-Noguer, F. GANimation: One-shot anatomically consistent facial animation. *Int. J. Comput. Vis.* **2020**, *128*, 698–713. [CrossRef]
21. Carrara, F.; Amato, G.; Brombin, L.; Falchi, F.; Gennaro, C. Combining GANs and autoencoders for efficient anomaly detection. In Proceedings of the 2020 25th International Conference on Pattern Recognition (ICPR), Milan, Italy, 10–15 January 2021; pp. 3939–3946.
22. Arjovsky, M.; Chintala, S.; Bottou, L. Wasserstein Generative Adversarial Networks. In Proceedings of the 34th International Conference on Machine Learning, Sydney, NSW, Australia, 6–11 August 2017; Volume 70, pp. 214–223.
23. Wang, B.; Wu, X.; Tang, Y.; Ma, Y.; Shan, Z.; Wei, F. Frequency domain filtered residual network for deepfake detection. *Mathematics* **2023**, *11*, 816. [CrossRef]
24. Gong, L.Y.; Li, X.J. A Contemporary Survey on Deepfake Detection: Datasets, Algorithms, and Challenges. *Electronics* **2024**, *13*, 585. [CrossRef]
25. Sandotra, N.; Arora, B. A comprehensive evaluation of feature-based AI techniques for deepfake detection. *Neural Comput. Appl.* **2023**, *36*, 1–29. [CrossRef]
26. Oshea, K.; Nash, R. An introduction to convolutional neural networks. *arXiv* **2015**, arXiv:1511.08458.
27. Sherstinsky, A. Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. *Phys. Nonlinear Phenom.* **2020**, *404*, 132306. [CrossRef]
28. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Columbus, OH, USA, 23–28 June 2014; pp. 580–587.
29. Juefei-Xu, F.; Wang, R.; Huang, Y.; Guo, Q.; Ma, L.; Liu, Y. Countering Malicious DeepFakes: Survey, Battleground, and Horizon. *Int. J. Comput. Vis.* **2022**, *130*, 1678–1734. [CrossRef] [PubMed]
30. Peng, C.L.; Gao, X.B.; Wang, N.N. Deep visual identity forgery and detection (in Chinese). *Sci. Sin. Inform.* **2021**, *51*, 1451–1474. [CrossRef]
31. Awesome Face Forgery Generation and Detection. Available online: https://github.com/clpeng/Awesome-Face-Forgery-Generation-and-Detection (accessed on 1 November 2024).
32. Preeti; Kumar, M.; Sharma, H.K. A GAN-Based Model of Deepfake Detection in Social Media. *Procedia Comput. Sci.* **2023**, *218*, 2153–2162. [CrossRef]
33. Yang, X.; Li, Y.; Qi, H.; Lyu, S. Exposing GAN-synthesized faces using landmark locations. In Proceedings of the ACM Workshop on Information Hiding and Multimedia Security, Paris, France, 3–5 July 2019; pp. 113–118.
34. Matern, F.; Riess, C.; Stamminger, M. Exploiting Visual Artifacts to Expose Deepfakes and Face Manipulations. In Proceedings of the 2019 IEEE Winter Applications of Computer Vision Workshops (WACVW), Waikoloa, HI, USA, 7–11 January 2019; pp. 83–92.
35. Guo, H.; Hu, S.; Wang, X.; Chang, M.C.; Lyu, S. Eyes tell all: Irregular pupil shapes reveal gan-generated faces. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Singapore, 22–27 May 2022; pp. 2904–2908.
36. Johnson, M.K.; Farid, H. Exposing digital forgeries through specular highlights on the eye. In Proceedings of the Information Hiding: 9th International Workshop, IH 2007, Saint Malo, France, 11–13 June 2007; Revised Selected Papers; Volume 9, pp. 311–325.
37. Wang, W. Evolution of StyleGAN3. In Proceedings of the 2022 International Conference on Electronics and Devices, Computational Science (ICEDCS), Marseille, France, 20–22 September 2022; pp. 5–13.
38. Gragnaniello, D.; Cozzolino, D.; Marra, F.; Poggi, G.; Verdoliva, L. Are GAN generated images easy to detect? A critical analysis of the state-of-the-art. In Proceedings of the 2021 IEEE International Conference on Multimedia and Expo (ICME), Virtual, 5–9 July 2021; pp. 1–6.
39. Khoo, B.; Phan, R.C.W.; Lim, C.H. Deepfake attribution: On the source identification of artificially generated images. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* **2022**, *12*, e1438. [CrossRef]

40. Arora, S.; Ge, R.; Liang, Y.; Ma, T.; Zhang, Y. Generalization and Equilibrium in Generative Adversarial Nets (GANs). In Proceedings of the 34th International Conference on Machine Learning, Sydney, NSW, Australia, 6–11 August 2017; Volume 70, pp. 224–232.
41. Yang, H.; Weinan, E. Generalization error of GAN from the discriminator's perspective. *Res. Math. Sci.* **2022**, *9*, 8. [CrossRef]
42. Marra, F.; Gragnaniello, D.; Verdoliva, L.; Poggi, G. Do GANs leave artificial fingerprints? In Proceedings of the 2019 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR), San Jose, CA, USA, 28–30 March 2019; pp. 506–511.
43. Yu, N.; Davis, L.S.; Fritz, M. Attributing fake images to gans: Learning and analyzing gan fingerprints. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 7556–7566.
44. Ding, Y.; Thakur, N.; Li, B. Does a GAN leave distinct model-specific fingerprints? In Proceedings of the 32nd British Machine Vision Conference, Virtual, 22–25 November 2021; p. 22.
45. McCloskey, S.; Albright, M. Detecting GAN-generated imagery using saturation cues. In Proceedings of the 2019 IEEE International Conference on Image Processing (ICIP), Taipei, Taiwan, 22–25 September 2019; pp. 4584–4588.
46. Nataraj, L.; Mohammed, T.M.; Chandrasekaran, S.; Flenner, A.; Bappy, J.H.; Roy-Chowdhury, A.K.; Manjunath, B.S. Detecting GAN generated fake images using co-occurrence matrices. *arXiv* **2019**, arXiv:1903.06836. [CrossRef]
47. Li, H.; Li, B.; Tan, S.; Huang, J. Identification of deep network generated images using disparities in color components. *Signal Process.* **2020**, *174*, 107616. [CrossRef]
48. Frank, J.; Eisenhofer, T.; Schönherr, L.; Fischer, A.; Kolossa, D.; Holz, T. Leveraging frequency analysis for deep fake image recognition. In Proceedings of the 37th International Conference on Machine Learning, Virtual, 13–18 July 2020; pp. 3247–3258.
49. Lewis, J.K.; Toubal, I.E.; Chen, H.; Sandesera, V.; Lomnitz, M.; Hampel-Arias, Z.; Prasad, C.; Palaniappan, K. Deepfake video detection based on spatial, spectral, and temporal inconsistencies using multimodal deep learning. In Proceedings of the 2020 IEEE Applied Imagery Pattern Recognition Workshop (AIPR), Washington, DC, USA, 13–15 October 2020; pp. 1–9.
50. Le, B.M.; Woo, S.S. Exploring the Asynchronous of the Frequency Spectra of GAN-generated Facial Images. *arXiv* **2021**, arXiv:2112.08050.
51. Li, M.; Zhong, Q.; Zhang, L.Y.; Du, Y.; Zhang, J.; Xiang, Y. Protecting the intellectual property of deep neural networks with watermarking: The frequency domain approach. In Proceedings of the 2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), Guangzhou, China, 29 December 2020–1 January 2021; pp. 402–409.
52. Wang, R.; Xu, J.F.; Ma, L.; Xie, X.; Huang, Y.; Wang, J.; Liu, Y. FakeSpotter: A Simple yet Robust Baseline for Spotting AI-Synthesized Fake Faces. In Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (ICJAI-20), Yokohama, Japan, 11–17 July 2020; pp. 3444–3451.
53. Dzanic, T.; Shah, K.; Witherden, F. Fourier spectrum discrepancies in deep network generated images. In Proceedings of the Advances in Neural Information Processing Systems 33 (NeurIPS 2020), Virtual, 6–12 December 2020.
54. Dharejo, F.A.; Deeba, F.; Zhou, Y.; Das, B.; Jatoi, M.A.; Zawish, M.; Du, Y.; Wang, X. TWIST-GAN: Towards Wavelet Transform and Transferred GAN for Spatio-Temporal Single Image Super Resolution. *ACM Trans. Intell. Syst. Technol. (TIST)* **2021**, *12*, 1–20. [CrossRef]
55. Xu, B.; Liu, J.; Liang, J.; Lu, W.; Zhang, Y. DeepFake Videos Detection Based on Texture Features. *Comput. Mater. Contin.* **2021**, *68*, 1375–1388. [CrossRef]
56. Thejas, N.U.; Nayak, H.D.; Siddique, A.B.; Danish, M.A.; Mamatha, H.R. Preprocessing and Feature Extraction based Deepfake Detection on Combined dataset. In Proceedings of the 2024 IEEE International Conference on Interdisciplinary Approaches in Technology and Management for Social Innovation (IATMSI), Gwalior, India, 14–16 March 2024; pp. 1–6.
57. Dalal, N.; Triggs, B. Histograms of oriented gradients for human detection. In Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA, 20–25 June 2005; pp. 886–893.
58. Tan, K.; Lee, W.S.; Gan, H.; Wang, S. Recognising blueberry fruit of different maturity using histogram oriented gradients and colour features in outdoor scenes. *Biosyst. Eng.* **2018**, *176*, 59–72. [CrossRef]
59. Padmashree, G.; Karunkar, A.K. Ensemble of Machine Learning Classifiers for Detecting Deepfake Videos using Deep Feature. *IAENG Int. J. Comput. Sci.* **2023**, *50*, 1279–1289.
60. Ricker, J.; Damm, S.; Holz, T.; Fischer, A. Towards the Detection of Diffusion Model Deepfakes. *arXiv* **2022**, arXiv:2210.14571.
61. Kanani, P.; Padole, M. Deep learning to detect skin cancer using google colab. *Int. J. Eng. Adv. Technol.* **2019**, *8*, 2176–2183. [CrossRef]
62. Canesche, M.; Bragança, L.; Neto, O.P.V.; Nacif, J.A.; Ferreira, R. Google Colab CAD4U: Hands-On Cloud Laboratories for Digital Design. In Proceedings of the 2021 IEEE International Symposium on Circuits and Systems (ISCAS), Daegu, Republic of Korea, 22–28 May 2021; pp. 1–5.
63. Gunawan, T.S.; Ashraf, A.; Riza, B.S.; Haryanto, E.V.; Rosnelly, R.; Kartiwi, M.; Janin, Z. Development of video-based emotion recognition using deep learning with Google Colab. *TELKOMNIKA (Telecommun. Comput. Electron. Control.)* **2020**, *18*, 2463–2471. [CrossRef]

64. Jia, L. Using Three GAN-Based Models to Provide Modeling Inspiration. Master's Thesis, Department of Computer Science, University of Reading, Reading, UK, 2022.
65. Liu, Z.; Luo, P.; Wang, X.; Tang, X. Deep learning face attributes in the wild. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 3730–3738.
66. Xu, Y.; Yayilgan, S.Y. When Handcrafted Features and Deep Features Meet Mismatched Training and Test Sets for Deepfake Detection. *arXiv* **2022**, arXiv:2209.13289.
67. Nickabadi, A.; Fard, M.S.; Farid, N.M.; Mohammadbagheri, N. A comprehensive survey on semantic facial attribute editing using generative adversarial networks. *arXiv* **2022**, arXiv:2205.10587.
68. Nightingale, S.J.; Farid, H. AI-synthesized faces are indistinguishable from real faces and more trustworthy. *Proc. Natl. Acad. Sci. USA* **2022**, *119*, e2120481119. [CrossRef] [PubMed]
69. Do, N.T.; Na, I.S.; Kim, S.H. Forensics Face Detection From GANs Using Convolutional Neural Network. In Proceedings of the 2018 International Symposium on Information Technology Convergence (ISITC 2018), Seoul, Republic of Korea, 12–14 November 2018; pp. 376–379.
70. Mo, H.; Chen, B.; Luo, W. Fake faces identification via convolutional neural network. In Proceedings of the ACM Workshop on Information Hiding and Multimedia Security, Innsbruck, Austria, 20–22 June 2018; pp. 43–47.
71. Marra, F.; Saltori, C.; Boato, G.; Verdoliva, L. Incremental learning for the detection and classification of gan-generated images. In Proceedings of the IEEE International Workshop on Information Forensics and Security (WIFS), Reykjavik, Iceland, 2–5 December 2019.
72. Tang, G.; Sun, L.; Mao, X.; Guo, S.; Zhang, H.; Wang, X. Detection of GAN-Synthesized Image Based on Discrete Wavelet Transform. *Secur. Commun. Netw.* **2021**, *2021*, *5511435*. [CrossRef]