ForensicsForest Family: A Series of Multi-Scale Hierarchical Cascade Forests for Detecting GAN-Generated Faces

Jiucui Lu, Jiaran Zhou, Junyu Dong, *Member, IEEE*, Bin Li, *Senior Member, IEEE*, Siwei Lyu, *Fellow, IEEE*, and Yuezun Li, *Member, IEEE*

***Abstract*— The prominent progress in generative models has significantly improved the authenticity of generated faces, raising serious concerns in society. To combat GAN-generated faces, many countermeasures based on Convolutional Neural Networks (CNNs) have been spawned due to their strong learning capabilities. In this paper, we rethink this problem and explore a new approach based on forest models instead of CNNs. Concretely, we describe a simple and effective forest-based method set, termed *ForensicsForest Family*, to detect GAN-generate faces. The ForensicsForest family is composed of three variants: *ForensicsForest*, *Hybrid ForensicsForest* and *Divide-and-Conquer ForensicsForest*. ForenscisForest is a novel Multi-scale Hierarchical Cascade Forest that takes appearance, frequency, and biological features as input, hierarchically cascades different levels of features for authenticity prediction, and employs a multi-scale ensemble scheme to consider different levels of information comprehensively for further performance improvement. Building upon ForensicsForest, we create Hybrid ForensicsForest, an extended version that integrates the CNN layers into models, to further enhance the efficacy of augmented features. Furthermore, to reduce memory usage during training, we introduce Divide-and-Conquer ForensicsForest, which can construct a forest model using only a portion of training samplings. In the training stage, we train several candidate forest models using the subsets of training samples. Then, a ForensicsForest is assembled by selecting suitable components from these candidate forest models. Our method is validated on state-of-the-art GAN-generated face datasets and compared with several CNN models, demonstrating the surprising effectiveness of our method in detecting GAN-generated faces.**

***Index Terms*— Digital forensics, GAN-generated face detection, random forest.**

I. Introduction

FACE forgery has significantly advanced in quality and efficiency, thanks to the advent of deep generative models (*e.g.*, GAN [1], [2], [3], VAE [4]). As shown in Fig. 1, the GAN-generated faces exhibit a high level of realism, which can hardly be distinguished by human eyes. Since human faces are important biometrics, the abuse of GANs can raise a severe security concern for society, *e.g.*, forging a fake identity on social platforms, deceiving users for fraud, etc [5]. As such, detecting the GAN-generated faces is of great importance.

The current GAN-generated faces detection methods [6], [7], [8] are mainly based on Convolutional Neural Network (CNN) models for their powerful learning abilities demonstrated in various vision tasks. With the availability of large-scale datasets of face forgeries [9], [10], it is possible to design new complex forms of CNN architectures with more parameters, without the risk of overfitting. However, despite these CNN-based methods having shown promising performance, they have two significant limitations that may obstruct their application in daily practice: 1) High demand for computing resources. Since the CNN models usually contain plenty of weight parameters, training them requires careful fine-tuning and expensive computing resources, *e.g.*, GPUs; 2) Security concerns. It has been proven that CNN-based methods are vulnerable to adversarial attacks, which can mislead the prediction by only adding imperceptible noises to input faces [11], [12]. It works because a large number of weight parameters makes the classification boundary more complicated, increasing the possibility of pushing a sample crossing the border with less effort. Also, due to the differentiability of CNNs, the attacks can be easily achieved by optimizing an objective function. These limitations drive us to think can the models being low resource demand and non-differentiable be used to detect GAN-generated faces?

In this paper, we rethink this problem and describe a simple and effective method set called *ForensicsForest Family* to expose GAN-generated faces (see Fig. 2). We adopt the forest model as the base in replace of CNN models to overcome the aforementioned limitations, as the forest is decision-based, which contains few weight parameters, and is not differentiable, thus naturally resisting a set of general adversarial attacks [11], [12]. The proposed ForensicsForest

Manuscript received 23 December 2023; revised 19 March 2024; accepted 23 April 2024. Date of publication 29 April 2024; date of current version 7 May 2024. The work of Jiaran Zhou was supported in part by NSFC under Grant 62102380, in part by NSF of Shandong Province under Grant ZR2021QF095, and in part by the China Postdoctoral Science Foundation under Grant 2021M693022. The work of Bin Li was supported in part by NSFC under Grant U23B2022. The work of Siwei Lyu was supported in part by NSF under Grant SaTC-2153112. The work of Yuezun Li was supported in part by China Postdoctoral Science Foundation under Grant 2021TQ0314 and Grant 2021M703036 and in part by the Fundamental Research Funds for the Central Universities. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Husrev T. Sencar. *(Corresponding authors: Jiaran Zhou; Yuezun Li.)*

Jiucui Lu, Jiaran Zhou, Junyu Dong, and Yuezun Li are with the College of Computer Science and Technology, Ocean University of China, Qingdao 266100, China (e-mail: lujiucui@stu.ouc.edu.cn; zhoujiaran@ouc.edu.cn; dongjunyu@ouc.edu.cn; liyuezun@ouc.edu.cn).

Bin Li is with Guangdong Key Laboratory of Intelligent Information Processing and Shenzhen Key Laboratory of Media Security, Shenzhen University, Shenzhen 518060, China (e-mail: libin@szu.edu.cn).

Siwei Lyu is with the Department of Computer Science and Engineering, University at Buffalo–SUNY, Buffalo, NY 14260 USA (e-mail: siweilyu@bufflao.edu).

Digital Object Identifier 10.1109/TIFS.2024.3395013

Fig. 1. Examples of real and GAN-generated faces (selected from StyleGAN2 [3]).

family is composed of three variants, which are *ForensicsForest*, *Hybrid ForensicsForest* and *Divide-and-Conquer ForensicsForest* respectively. ForensicsForest is a newly proposed architecture called Multi-scale Hierarchical Cascade Forest, containing three main components: Input Feature Extraction, Hierarchical Cascade Forest, and Multi-scale Ensemble, respectively (see Fig. 3). The input feature extraction is a preprocessing step to extract informative features with fixed dimensions, enabling our detector to be independent of image size. Concretely, we first split the input image into multiple patches and then extract two types of features for each patch: the color histogram as appearance features and the power spectrum as frequency features. To capture the characteristics of faces, we additionally extract the facial landmarks as biological features. This is because the topology of the facial landmarks is typically not taken into account in the generation of faces using GANs, which can result in inconsistency with the facial landmarks found in real faces [13]. These features are concatenated and sent into a hierarchical cascade forest for prediction. The hierarchical cascade forest is composed of several cascade forest layers, where the features of each patch are hierarchically integrated into consecutive layers. This design allows for alternatively processing each patch, augmenting the features of each patch by incorporating the knowledge of the previous patch while minimizing the computational overhead. Moreover, we propose a multi-scale ensemble approach that takes features of different scales by adjusting the sizes of patches. This enables the model to effectively learn the discriminative features and combines the obtained results for final prediction.

Expanding on the concept of ForensicsForest, we propose Hybrid ForensicsForest as an extension that explores the feasibility of integrating CNN layers into the forest model. This is motivated by the proven effectiveness of CNN in learning tasks. By combining CNN with the forest model, we aim to enhance the output of forest layers and improve the effectiveness of augmented features. In addition, constructing forest-based models typically requires loading all training samples into memory simultaneously, leading to significant memory costs. As such, we further propose the Divide-and-Conquer ForensicsForest to address the issue of high memory cost while maintaining favorable detection performance. In this approach, we divide the training stage into two steps. Firstly, we train several candidate forest models using only a small portion of training samples. This allows us to reduce the memory requirement during training. Next, we develop a strategy to select the best components from these candidate forest models and assemble these components as a final forest model. This final model retains the favorable detection performance while significantly reducing the resource costs associated with using all training samples at once.

We conduct extensive experiments on various types of state-of-the-art GAN-generated faces, and compare our method with several counterparts, showing that our method is surprisingly effective in exposing GAN-generated faces in comparison to CNN-based methods. Moreover, we thoroughly study the effect of each component of our method, as well as the robustness and generalization ability.

The contribution of this paper is summarized as follows

1) We describe a new forest-based method named *ForensicsForest*, which is a novel Multi-scale Hierarchical Cascade Forest to process face patches hierarchically in a multi-scale ensemble manner. To the best of our knowledge, we are the first to investigate the potential of forest models to expose GAN-generated faces, offering fresh insights for future research.
2) Building upon ForensicsForest, we develop *Hybrid ForensicsForest*, which explores the integration of CNN layers into forest layers, aiming to further enhance the effectiveness of augmented features.
3) To reduce the memory cost, we describe *Divide-and-Conquer ForensicsForest*, an approach that selects the most suitable components from a set of candidate forest models which are trained using a small portion of samples, and assemble these components as a final ForensicsForest.
4) We extensively evaluate our method on recent GAN-generated faces, including StyleGAN [14], StyleGAN2 [3], and StyleGAN3 [15]. We compare our method with CNN-based counterparts and thoroughly analyze the effect of various modules in our method. Furthermore, we validate the performance of our method on robustness, cross-datasets, and other generative models, such as ProGAN [1], StarGAN [16] and Diffusion model [17].

This paper is extended from our `ICME conference paper [18] (Oral)` in several aspects: (1) We improve the input feature extraction module by adding facial landmarks as biological features; (2) We propose Hybrid ForensicsForest, which integrates the CNN layers into ForensicsForest, and validate its performance on GAN-generated face datasets; (3) We then develop Divide-and-Conquer ForensicsForest, which selects the suitable components from a set of candidate forest models to form a final model, and investigate its performance with various settings; (4) We conduct more experiments on the comparison with dedicated GAN-generated face detection methods, and study the ability of our method on different datasets and different generative models.

II. Background and Related Works

A. GAN-Generated Faces Detection

In the early stage, many forensic methods are based on statistic signals to expose forged images [19], [20], [21]. However, with the advent of GAN models, the faces can be synthesized in an end-to-end fashion with very high visual

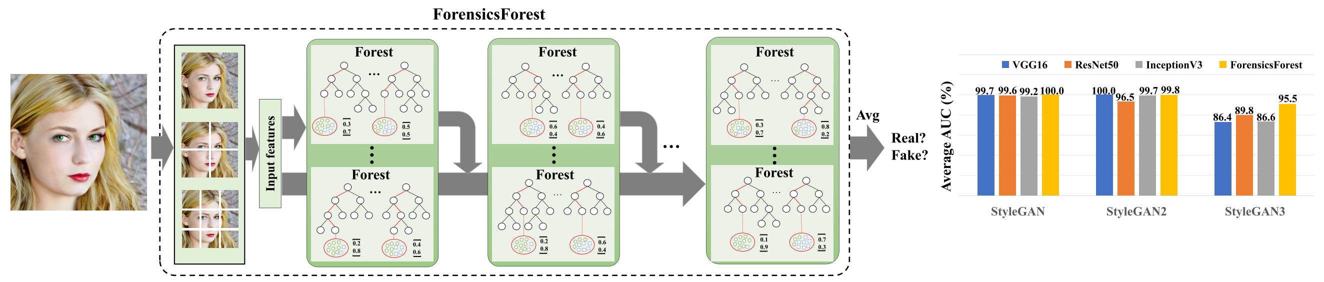

Fig. 2. Diagram of the proposed ForensicsForest, the core of *ForensicsForest Family* for detecting GAN-generated faces. Our method can achieve competitive and even better performance compared to CNN-based detection methods. Other two variants, Hybrid ForensicsForest and Divide-and-Conquer ForensicsForest are developed based on ForensicsForest. The details of each variant can be seen in Section III, Section IV, and Section V respectively.

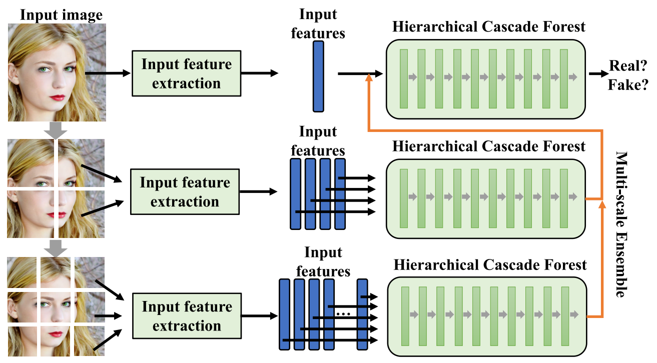

Fig. 3. Overview of the proposed multi-scale hierarchical cascade forest.

quality, and these early methods are no longer effective in detecting GAN-generated faces.

Recently, CNNs have been widely employed in detection methods due to their excellent performance on vision tasks, in order to expose subtle forgery traces. Some methods learn the detectable clues by directly training CNN models with vanilla or self-designed architectures in a supervised way, *e.g.*, [6], [7], [22], [23], [24], [25], [26], [27], and [28]. There are also many methods relying on empirically selected clues in the following aspects: 1) Physiological signals. Li et al. [29] propose a Long-term Recurrent Neural Network (LRCN) to detect the inconsistency in eye blinking. Yang et al. [30] expose the head pose inconsistency to expose generated faces. Guo et al. [8] find that the generated faces do not consider the consistency in eye corneal specular highlight and use it as clues to expose generated faces. Similarly, other physiological signals such as remote photoplethysmography (rPPG) are also used in [31] to exhibit the inconsistency between real and fake. 2) Artifact signals. Several works find that the upsampling process in GAN face generation can leave specific artifacts [32], [33], and the blending artifacts are found in face swapping operations [34], [35]. More recently, LGrad [36] utilize gradient artifacts as the general clues for detection. 3) Spatial or frequency signals. Several works detect the inconsistency in different color domains, *e.g.*, [37], [38], and [39], while other works attempt to extract traces in the frequency domain, *e.g.*, [40], [41], [42], and [43]. Extracting these clues relies on CNNs or manually designed modules, and then sending these clues into CNNs for final prediction. In contrast to these recent methods, we investigate the feasibility of using a non-CNN model and describe a new series of forest-based methods to expose GAN-generated faces.

B. Deep Forest

The forest is the classical decision model for classification. Due to the nature of decision models, they are not differentiable. In contrast, Deep Neural Networks (DNN) models are differentiable networks with deeper architectures and multiple layers of differentiable parameterized modules. Inspired by the success of DNN models, Deep Forest [44] is proposed in a non-DNN style deep model with a cascade structure. Specifically, Deep Forest is a layer-by-layer architecture and each layer contains multiple decision trees. The output of each layer is concatenated with the input and sent into the next layer. Due to this novel design, it performs better than the traditional decision models (*e.g.*, Random Forests, XGBoost, etc), and achieves competitive, sometimes better performance than CNN models. Compared to DNNs, in addition to the non-differentiable property, Deep Forest requires much fewer hyper-parameters, and its model complexity can be automatically determined in a data-dependent way. However, it usually focuses on the general classification in small scales (*e.g.*, CIFAR10 [45], MINIST [46]), which can hardly be directly applied to the task of GAN-generated face detection. The difference between our method and Deep Forest is elaborated in Section III-C.

III. FORENSICSFOREST

A. Model Architecture

Our method contains three components: Input Feature Extraction, Hierarchical Cascade Forest, and Multi-scale Ensemble. We introduce each component in a sequel.

1) Input Feature Extraction: Instead of using the whole image as input, we use the features extracted from the image. Note that the extracted features are in fixed dimensions, regardless of the input image size. Thus our method is feasible to handle the input images of arbitrary size. Specifically, we extract three types of features: appearance features, frequency features, and biology features.

For appearance features, we employ simple color histograms. Concretely, we first make a color histogram for each channel and flatten them together as the appearance features (3×256). Moreover, we extract frequency features as complementary to appearance features. We first convert each channel of the input image into a frequency map using the Fast Fourier Transform (FFT) and then transform the frequency map into a power spectrum using the azimuthal average as the frequency features (3×88). For biology features,

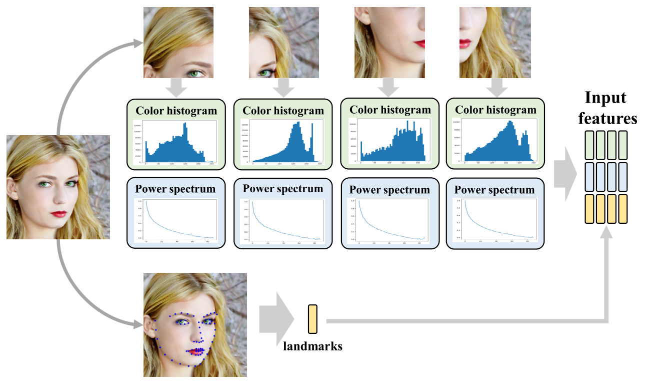

Fig. 4. Illustration of input feature extraction. We extract the color histogram, power spectrum, and facial landmarks as the appearance, frequency, and biology features respectively. See text for details.

we extract the facial landmarks and use the coordinates of these landmarks to represent the characteristics of faces (2×68). Fig. 4 shows the overview of input feature extraction.

In a general formulation, we can extract appearance and frequency features from $N (N \geq 1)$ patches of input images, where $N = 1$ indicates the two features are extracted from the whole image, and extract biology features from the whole image.

2) Hierarchical Cascade Forest: Given the extracted input feature vectors, we develop a hierarchical cascade forest to determine the authenticity. As shown in Fig. 5, this forest contains multiple hierarchical cascade blocks, where each block is composed of N cascade forest layers. Each layer contains two random forests and two completely random forests. The random forest is composed of CART decision trees which are created using the Gini index to select the best feature attributes for a partition. By contrast, the completely random forest consists of completely random trees created by randomly selecting feature attributes. In our task, we have two classes to predict, *i.e.*, real or fake. Thus each forest generates a two-dimension probability vector. The concatenation of these vectors inside a layer is the augmented feature. Specifically, for each random forest, we use 100 random trees and the same setting for each completely random forest, which contains 100 completely random trees. The workflow is as follows: the output of i-th layer is the augmented feature, which is then concatenated with the feature vector of $(i + 1)$-th patch as the input of $(i + 1)$-th layer. For the first block, the input of the 1-st layer is the input feature of the 1-st patch. For other blocks, the input of the 1-st layer is the concatenation of the input feature of the 1-st patch and the augmented feature from the last block.

3) Multi-Scale Ensemble: The output of the hierarchical cascade forest can be used for the final prediction. However, only relying on one hierarchical cascade forest overlooks the forgery information in other scales. Hence, we propose a multi-scale ensemble, which incorporates the forgery information of multiple scales as the final feature for prediction. As shown in Fig. 3, we construct several hierarchical cascade forests and each forest corresponds to one patch number (*e.g.*, $N = 1, 4, 9$). Specifically, we use the forest for the whole image ($N = 1$) as the base forest. The input features from this forest are concatenated with the augmented features from other forests ($N > 1$) as the final input. By considering multiple scales, the detection performance is further improved.

B. Training and Inference

1) Training: Different from CNN models, our method is constructed on decision trees. Thus the training process is not to learn parameters, but to create the forest structure directly on given training samples. Note that our method contains two types of forests, random forest and completely random forest. Denote the training set as $\mathcal{D}$, and $(x, y) \in \mathcal{D}$ as a training sample. Note that $y \in \{0, 1\}$, where 0 denotes real and 1 denotes fake. The Gini index of $\mathcal{D}$ is defined as

$$\text{Gini}(\mathcal{D}) = 2p(1 - p), \tag{1}$$

where p is the probability of samples being labeled 1. Let $\mathcal{A}_i$ be an attribute from feature $\mathcal{A}$ that has possible values in set $\mathcal{V}$. Assume $a \in \mathcal{V}$ is one of the possible values. The Gini index of $\mathcal{A}_i = a$ can be defined as

$$\text{Gini}(\mathcal{D}, \mathcal{A}_i = a) = \frac{|\mathcal{D}_1|}{|\mathcal{D}|}\text{Gini}(\mathcal{D}_1) + \frac{|\mathcal{D}_2|}{|\mathcal{D}|}\text{Gini}(\mathcal{D}_2),$$
$$\mathcal{D}_1 = \{(x, y) \in \mathcal{D} | \mathcal{A}_i(x) = a\}, \mathcal{D}_2 = \mathcal{D} - \mathcal{D}_1, \tag{2}$$

The creation of decision trees is to find the best partition recursively, as

$$\underset{a \in \mathcal{V}, \mathcal{A}_i \in \mathcal{A}}{\arg\min}\ \text{Gini}(\mathcal{D}, \mathcal{A}_i = a). \tag{3}$$

These forests randomly select $\sqrt{d}$ candidate feature attributes, where d is the number of feature attributes. To mitigate the risk of overfitting, each forest is created using k-fold cross-validation, that is first to randomly divides $\mathcal{D}$ into k sets without overlap, and then use $k - 1$ sets for training and rest set for validation. This process is repeated k times until each set has been used for validation.

In contrast to CNN models, we dynamically decide the number of cascade layers in ForensicsForest. The criterion for terminating the training process is as follows: 1) Meeting the maximum number of layers. In order to control resource consumption, the forest stops growing if the number of layers is greater than the maximum, or 2) The performance is not improved anymore even if more layers are added. Specifically, we constantly evaluate the output of each layer by feeding training data. If a predefined number of layers behind the current layer can not further improve the performance, we do not add these layers into the forest and terminate the training process.

2) Inference: Once the forest is constructed, we can send testing face images to it and average the output of each forest of the last layer to obtain the final prediction.

C. Comparing With Classic Deep Forest

1) Input Feature Extraction: The classic Deep Forest is designed for low-dimensional images, *e.g.*, MINIST (28×28), CIFAR (64×64) dataset, which can not handle the high-dimensional images such as GAN-generated face images, *e.g.*, StyleGAN (1024×1024), due to the significant high

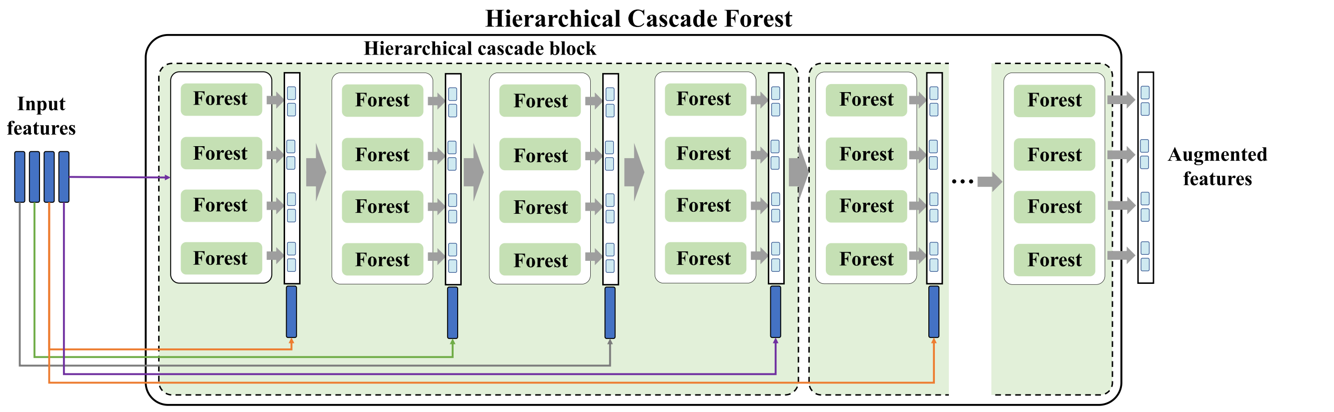

Fig. 5. Overview of the hierarchical cascade forest. The input features are hierarchically sent into the corresponding layer in a hierarchical cascade block. See text for details.

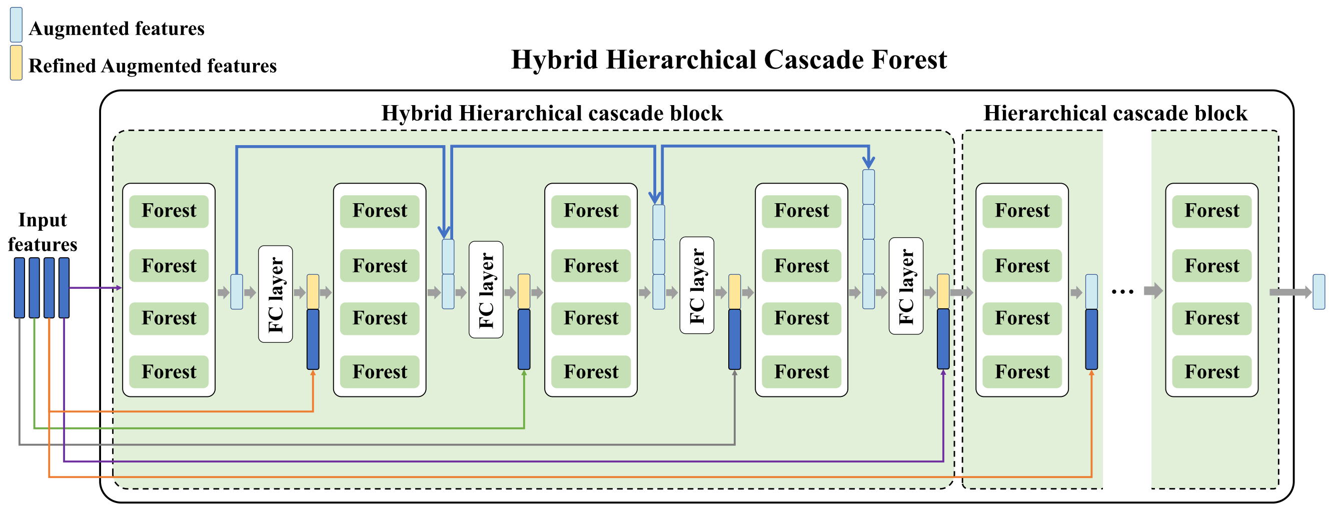
Augmented features
Refined Augmented features
Hybrid Hierarchical Cascade Forest
Hybrid Hierarchical cascade block
Hierarchical cascade block
Input features
Forest
FC layer

Fig. 6. Overview of the proposed Hybrid ForensicsForest. In contrast to ForensicsForest, Hybrid ForensicsForest integrates FC layers into the model, appending a FC layer behind the augmented features from the forest layer and refining the features for the next layer.

resource consumption. Thus, we convert the GAN-generated face images into low-dimensional features as the input. The advantages are that the resource consumption is independent of the dimension of images, and redundant content can be discarded.

2) Hierarchical Cascade Forest: The classic Deep Forest uniformly cascades the input features with the augmented features out from each layer. Thus the computation cost is positively correlated with the size of features. In contrast, we propose a hierarchical cascade, which first splits the input features into different pieces (*e.g.*, 4 pieces in Fig. 5), and alternatively cascades different pieces with the augmented features out from each layer. In this way, the computation cost is greatly reduced, and more importantly, the augmented features of one layer can absorb the knowledge of the previous piece of feature, which can learn local associations between pieces while obtaining the global feature ultimately.

3) Multi-Scale Ensemble: The classic Deep Forest does not consider the multi-scale process. However, since the GAN-generated face images are usually in high dimensions containing more complex content, only using one scale is ineffective. Thus we propose a multi-scale ensemble, which considers different scales of information, by fusing the augmented features from different scales together for the final prediction.

IV. Hybrid ForensicsForest

The effectiveness of CNNs in various tasks has been well-established, mainly due to their learnable parameters. Inspired by this, we aim to investigate the potential of integrating these learnable parameters into the forest model. This integration seeks to strike a balance between CNNs and decision-based models. Building on the proposed ForensicsForest, we describe Hybrid ForensicsForest, which integrates the CNN layers into the forest model layers, to further refine the augmented features in ForensicsForest.

In our approach, we employ the Fully Connected (FC) layer to enhance the forest features. As shown in Fig. 6, we append a FC layer after each forest layer and send the refined augmented features into the next layer. We opt for the FC layer instead of the Convolution layer because the augmented features derived from the forest are compact with low dimensions, making them more amenable to global refinement. To maximize the potential of the FC layer, we propose an iterative cascade strategy, which stacks the augmented features of the previous layer to the current layer (Blue arrows in Fig. 6). This allows

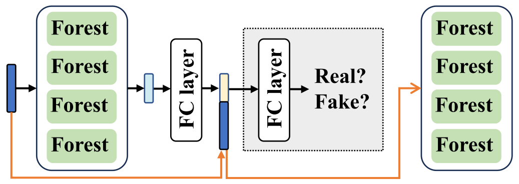

Fig. 7. Illustration of the training process of Hybrid ForensicsForest. In training, we add an extra FC layer (shown in the grey box) to implicitly guide the learning. In testing, we only use the first FC layer.

for the collection of more information to be used for the FC layer. More strategies involving the use of FC layers can be found in Section VI-C.

Training this Hybrid ForensicsForest is non-trivial, as it involves combining a decision model with learnable parameters. This makes it challenging to optimize the model in an end-to-end fashion using the existing training schemes for ForensicsForest. To address this challenge, we develop a hybrid training scheme that can jointly construct the forests and learn the parameters of FC layers. Specifically, we first construct the forest layer using the training scheme described in Section III-B. Then we train the appended FC layers. However, training the FC layer within the forest model poses difficulties as the gradients fail to propagate through the model. Consequently, it is not feasible to optimize the FC layer using a task-related objective function (*e.g.*, binary classification) based on the input and output of the model. To overcome this limitation, we train the FC layer in each forest layer separately. We utilize the augmented features from the forest layer as training samples for the FC layer and introduce a second FC layer as an auxiliary component to guide the learning of the first FC layer. The objective for training these two FC layers is a simple binary cross-entropy loss, *i.e.*, distinguishing between real and fake. In this way, the first FC layer learns to align the augmented features towards a distinguishable distribution. Once training is complete, we retain only the first FC layer and disable the second FC layer during the testing phase. The training procedure is depicted in Fig. 7.

Note that we do not directly add the FC layers on images. Instead, we insert them in an alternative manner among the forest layers. Despite the FC layers being differentiable, the gradients cannot be back-propagated to the input image due to the inherent nature of forests. Thus the proposed Hybrid ForensicsForest remains resilient against general adversarial attacks.

V. Divide-and-Conquer ForensicsForest

Unlike the batch-style training scheme commonly used in CNN, ForensicsForest requires loading all training samples into memory at once to construct the forest structure. However, this training scheme poses limitations when applied in the wild, as the ForensicsForest model may fail to be constructed if the available memory size is insufficient to accommodate all training samples. To address this issue, one possible solution is to only use a portion of the training samples. However, this approach inherently lacks the knowledge from the excluded training samples. To overcome this issue, we propose Divide-and-Conquer ForensicsForest which can greatly reduce the memory cost while retaining the favorable detection performance.

In the training stage, we randomly divide the training samples into two portions and use one portion to train a ForensicsForest. By utilizing only a part of the training samples, the memory required for constructing the forest is significantly reduced. This training process is repeated several times, resulting in a collection of forest models, known as candidate forest models. Since each forest model is trained using a randomly divided portion of training samples, it can capture a specific perspective knowledge of all training samples. We believe that by carefully designing a strategy, we can integrate these different perspectives of knowledge from the forest models into a final one. This integration aims to be equivalent to using all training samples at once. This strategy is elaborated as follows.

Denote the whole training set as $\mathcal{D}$ and the repeated number as T. At the i-th time, we randomly divide $\mathcal{D}$ into two portions, $\mathcal{D}_1^i$ and $\mathcal{D}_2^i$, using a ratio $r = \frac{|\mathcal{D}_1^i|}{|\mathcal{D}|}$ to control the size of each portion. We design a candidate ForensicsForest with $m (m \geq 4)$ forests in a layer and train it using $\mathcal{D}_1^i$. After training, we need to select suitable components, *i.e.*, forests from the candidate model, and assemble them to form a final ForensicsForest. To take into account both fitting and generalization, we assess the effectiveness of each forest in a layer by computing its average accuracy on $\mathcal{D}_1^i$ and $\mathcal{D}_2^i$. Then we select top-n forests in a layer having the best average accuracy and assemble these forests into a layer to construct a ForensicsForest at i-th time. To be consistent with our ForensicsForst configuration, n is set to 4 which contains two random forests and two completely random forests respectively. This process is repeated for T times and constructs a candidate forest model for each time. Finally we ensemble these forest models and average the output of each model as the final prediction. By doing so, the memory load of constructing a ForensicsForest can be divided into several small pieces, which significantly reduces the memory cost by tuning with a ratio r. The procedure is illustrated in Fig. 8.

VI. Experiments

A. Experimental Settings

1) Datasets: Our method is validated on three types of GAN-generate face images, which are StyleGAN [14], StyleGAN2 [3] and StyleGAN3 [15] respectively. The face quality is improved along with the version of StyleGAN increasing. The real face images are collected from the Flickr-Faces-HQ (FFHQ) dataset [14]. For StyleGAN and StyleGAN2 dataset, we randomly select $5,000$ images from FFHQ and StyleGAN or StyleGAN2 faces respectively. StyleGAN3 only provides 867 images. Thus we randomly select the same amount of real images from FFHQ to construct the StyleGAN3 dataset. The ratio of training and testing set is $8:2$ for all datasets and all images have the same resolution of 1024×1024.

2) Implementation Details: The proposed ForensicsForest is trained and tested only using an Intel Core-i5 12400F CPU and no data augmentation is used. The number of forests in a layer is set to 4 and the number of trees in a forest

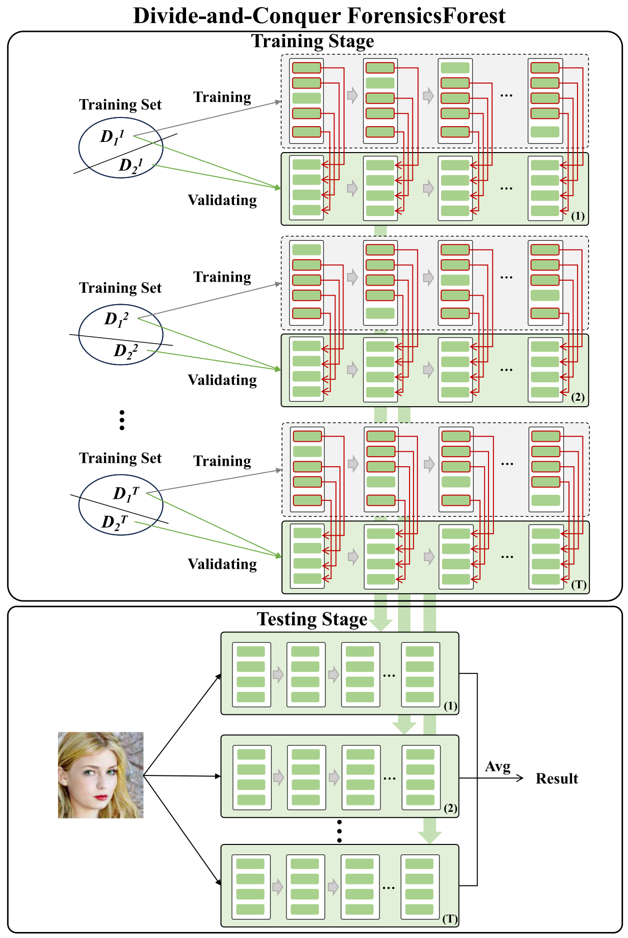

Fig. 8. Overview of the training (top) and testing process (bottom) in Divide-and-Conquer ForensicsForest. In training, for each time, we use $\mathcal{D}_1^i$ to construct a primitive model and use $\mathcal{D}_1^i$ and $\mathcal{D}_2^i$ to select suitable components and assemble them as a final model. In testing, we use these assembled models and average the predictions as the final result. See text for details.

is set to 100. We use four scales as $N = 1, 2, 3, 4$, where $N = 2, 3$ denotes to vertically split the input image into 2 and 3 patches. In training, the maximum number of layers is set to 20 and the number of layers for checking the performance improvement is set to 2. The maximum training epoch for each CNN model is set to 20. All CNN models are trained using a single Nvidia 2080ti GPU. For Hybrid ForensicsForest, we set the learning rate to 0.01, and other basic settings are the same as ForensicsForest. For Divide-and-Conquer ForensicsForest, the ratio r of training and validating is set to 0.9, the repeated time number T is set to 5, and the number m of forests in each candidate forest model is set to 16.

B. Results of ForensicsForest

Compared to CNN Models: To demonstrate the efficacy of our method, we compare our method with several mainstream CNN models, which are VGG16 [47], ResNet18 [48], ResNet50 [48], Inception V3 [49], MobileNet [50], ResNeXt [51], MNASNet [52], EfficientNet [53], and RegNet [54] respectively. We load their pre-trained weights on ImageNet and fine-tune all layers of them on each StyleGAN dataset. The learning rate is set to 0.001. The performance of our method in comparison to CNN models is shown in Table I (Top). The evaluation metrics are Accuracy (ACC) and Area Under Curve (AUC) following previous works. We can see for all of these datasets, our method achieves competitive and even better performance than CNN models. In particular, our method outperforms all CNN models by at least 4.7% in ACC on the StyleGAN dataset. For example, our method surpasses ResNet50 by 6.8% in ACC and outperforms MNASNet by a large margin, 13.3%, in ACC. Our method also performs very well on the StyleGAN2 dataset, which achieves 98.7% in ACC and 99.8% in AUC. StyleGAN3 dataset is more challenging as the synthesis quality is the best and the training set is small. Thus all methods perform compromised compared to other datasets. But it can be seen our method can still achieve the best performance on this dataset, which outperforms others by around 15% in ACC and 11.3% in AUC on average, demonstrating the effectiveness of our method on detecting GAN-generated face images.

TABLE I
PERFORMANCE (%) OF DIFFERENT METHODS ON THREE DATASETS

Method	**StyleGAN**		**StyleGAN2**		**StyleGAN3**	
	ACC	AUC	ACC	AUC	ACC	AUC
VGG16 [47]	93.5	99.7	97.8	100.0	72.3	86.4
ResNet18 [48]	92.2	98.9	86.1	96.1	70.2	85.3
ResNet50 [48]	93.2	99.6	90.1	96.5	81.5	89.8
Inception V3 [49]	94.6	99.2	97.8	99.7	77.5	86.6
MobileNet [50]	94.6	98.3	94.6	98.8	71.7	81.6
ResNeXt [51]	95.3	99.0	86.1	94.8	74.0	82.2
MNASNet [52]	86.7	97.0	81.1	97.0	64.7	71.1
EfficientNet [53]	90.3	96.9	92.1	97.7	74.1	89.3
RegNet [54]	92.2	99.7	93.4	99.7	74.6	85.7
CNNDetection [6]	98.4	99.8	97.9	99.7	78.9	95.1
GramNet [7]	99.6	100.0	97.6	99.8	70.0	89.7
CoALBP+LPQ [22]	93.8	98.3	97.4	99.6	60.3	64.4
Patch-forensics [24]	98.1	99.7	**99.4**	**100.0**	62.9	71.7
AutoGAN [32]	93.9	99.2	92.2	98.1	77.7	93.5
Meso4 [55]	80.3	93.0	60.5	82.6	53.5	56.3
MesoInception [55]	86.0	96.8	91.4	98.3	63.5	80.0
Xception [9]	97.5	99.9	98.0	99.9	76.8	84.7
EfficientB4 [53]	93.0	99.8	94.6	99.9	69.4	85.6
Capsule [56]	96.4	100.0	94.4	100.0	73.3	89.4
FWA [34]	98.5	100.0	98.8	100.0	79.4	87.0
Face X-ray [35]	97.6	99.9	97.8	100.0	75.7	88.5
FFD [57]	98.6	100.0	98.9	100.0	80.2	89.0
CORE [58]	98.7	100.0	98.5	100.0	78.4	87.1
Recce [59]	98.4	100.0	99.0	100.0	79.8	85.9
UCF [60]	95.1	99.2	97.8	99.8	76.2	85.1
F3Net [41]	98.5	100.0	99.0	100.0	80.2	89.4
SPSL [43]	97.6	99.8	98.0	99.7	76.2	87.5
SRM [61]	98.7	100.0	98.7	100.0	77.6	87.6
Ours	**100.0**	**100.0**	98.7	99.8	**88.4**	**95.5**

Compared to GAN-Generated Face Detection Methods: We also compare our method with five recent CNN-based GAN-generated face detection methods, CNNDetection [6] and GramNet [7], CoALBP+LPQ [22], Patch-forensics [24] and AutoGAN [32]. For these methods, we retrain them on our datasets using their released codes. The results are shown in Table I (Middle). We can observe that the compared five methods show good performance on the StyleGAN dataset. However, the performance of these methods drops on the StyleGAN3 dataset, especially at the ACC metric.

TABLE II
TRAINING TIME (SECONDS) OF DIFFERENT METHODS

Method	StyleGAN	StyleGAN2	StyleGAN3
VGG16 [47]	3390.0s	3486.8s	229.5s
ResNet50 [48]	2879.1s	2646.7s	212.6s
Inception V3 [49]	3745.3s	3540.0s	350.6s
Ours	**232.5s**	**224.1s**	**142.6s**

TABLE III
PERFORMANCE (%) OF DIFFERENT METHODS ON FACES GENERATED BY PROGAN, STARGAN, AND LDM

Method	ProGAN		StarGAN		LDM	
	ACC	AUC	ACC	AUC	ACC	AUC
VGG16 [47]	**99.0**	99.9	96.0	99.6	99.5	100.0
ResNet50 [48]	97.0	99.5	93.0	97.5	99.5	100.0
Inception V3 [49]	87.5	96.0	95.5	99.9	90.8	99.3
Ours	97.5	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**

In comparison, our method performs well and is stable in all datasets, which represents the effectiveness of our method.

Compared to Deepfake Detection Methods: Besides the GAN-generated face detection methods, we also compare our method with recent Deepfake detection methods, including Meso4 [55], MesoInception [55], Xception [9], EfficientB4 [53], Capsule [56], FWA [34], Face X-ray [35], FFD [57], CORE [58], Recce [59], UCF [60], F3Net [41], SPSL [43], SRM [61]. For a fair comparison, we retrain these methods on our datasets following the codes provided in [62]. As shown in Table I (Bottom), most of the methods perform well on StyleGAN and StyleGAN2 datasets except Meso4. However, when tested on the StyleGAN3 dataset, there is a notable performance drop. This trend is consistent with GAN-generated face detection methods in general, as these methods mainly focus on detecting Deepfake forgery types, that typically manipulate the local area rather than the whole image in the GAN-generated faces. In contrast, our method outperforms these Deepfake detection methods by a large margin, particularly on the StyleGAN3 dataset.

Training Complexity: Table II records the time (seconds) of training corresponding methods. We compare our method with three classical architectures: VGG16, ResNet50, and Inception V3. This comparison is considered representative as these models are commonly used in recent counterpart methods, *e.g.*, [6], [34], [57], and [60]. It can be seen that despite CNN models being trained on GPU while our method is trained on CPU, our method has significantly lower time consumption than CNN models.

Performance on More Generative Models: We also validate our method on other generative models, such as ProGAN [1], StarGAN [16], and a recent diffusion-based model named LDM [17]. To construct these datasets, we run their official code under default settings. We generate 500 face images as a fake set and select the exact amount of images from the FFHQ dataset as a real set. The training and testing split is set to 8 : 2. The resolution of all images is 256×256. Table III shows the performance of our method and other CNN models to detect generated faces. It can be seen that our method achieves the best performance, almost 100% on ACC and 100% on AUC, surpassing all other CNN models. This result demonstrates that our method is effective in exposing the fake faces generated by other models.

Performance on Cross-Datasets: To rule out the bias in datasets, we conduct different cross-dataset experiments to validate the generalization ability of our method. Specifically, our method is trained on a dataset selected from StyleGAN, StyleGAN2, and StyleGAN3 and is tested on others. The results are shown in Table IV. We can observe that our method generally performs better than other CNNs, especially in the setting of training on StyleGANs and testing on StarGAN and LDM. Moreover, compared to GAN-generated face detection methods, our method also shows favorable and competitive performance. We attribute it to the proposed hierarchical and multi-scale learning scheme in the forest model.

Ablation Study: This part studies the effect of various settings of ForensicsForest, including the input features, different multi-scale ensemble strategies, and their effects respectively.

1) Effect of Each Component of Input Features: Table V shows the effect of each component and their combinations of the input feature vector. The color histogram, power spectrum, and facial landmarks used in our method are denoted as CH, PS, and FL respectively. The first three rows represent using a color histogram, power spectrum, and facial landmarks respectively. The other rows represent combinations of these feature components in different ways. It can be seen that using an arbitrary component can achieve a decent performance on all datasets, affecting exposing the GAN-generated faces. Among these components, we can observe that for StyleGAN3, the frequency feature, *i.e.*, power spectrum, performs best, and the biology feature, *i.e.*, facial landmarks, achieves the second best. It is because frequency and biology features are usually not used in training the GAN models, thus the generated faces tend to lack these clues correspondingly. Using two combinations of these components can improve the performance than only using a single one, which represents each component is dedicated to one aspect, and can be complementary with other each. The last row is our method that considers all of these components, which achieves the best performance than others, demonstrating the effectiveness of each component in this task.

2) Effect of Multi-Scale Ensemble: To validate the effectiveness of a multi-scale ensemble, we conduct experiments under two settings, which are with multi-scale (w/ ME) and without multi-scale (w/o ME). Table VI shows the performance of each case on three datasets. For (w/o ME), $N = 1$ denotes using the whole image as input, and $N = 4$ denotes using four image patches as input. For (w/ ME), $N = 1$ denotes integrating the augmented features of $N = 4$ patches into the input features of the whole image for prediction, and the setting is the same for $N = 4$, that is integrating the augmented features of the whole image into the input features of $N = 4$ patches for prediction. We can observe that using either the whole image or the four patches of the image can not reach the best performance, as it overlooks either the local or the global information. Especially for StyleGAN3, by using a multi-scale ensemble, the performance is further improved by 0.9% when $N = 1$ and 3.7% when $N = 4$ in ACC, as it considers both global and local information, which demonstrates its effectiveness in our method. Note that the

TABLE IV
PERFORMANCE OF DIFFERENT METHODS UNDER CROSS-DATASET SETTING

Method	Training	StyleGAN		StyleGAN2		StyleGAN3		ProGAN		StarGAN		LDM	
		ACC	AUC	ACC	AUC	ACC	AUC	ACC	AUC	ACC	AUC	ACC	AUC
VGG16 [47]	**StyleGAN**	93.5	99.7	80.1	92.2	50.6	59.3	50.5	58.4	50.1	68.2	52.3	56.7
ResNet50 [48]		93.2	99.6	74.6	88.6	51.7	63.6	**51.3**	66.7	51.1	63.2	52.5	68.8
Inception V3 [49]		94.6	99.2	68.3	84.7	53.8	65.2	50.9	66.7	50.3	56.1	51.5	58.4
GramNet [7]		99.6	100.0	53.4	96.2	50.0	**78.8**	50.5	**93.1**	50.0	79.4	50.0	63.4
Patch-forensics [24]		98.1	99.7	58.1	96.0	50.5	75.0	50.5	60.2	**60.7**	78.3	50.4	61.3
AutoGAN [32]		93.9	99.2	53.0	71.8	51.3	59.2	50.0	57.5	50.0	75.0	56.0	97.1
Ours		**100.0**	**100.0**	**96.2**	**99.5**	**67.6**	69.8	50.0	62.6	52.0	**82.5**	**71.8**	**99.2**
VGG16 [47]	**StyleGAN2**	56.0	77.1	97.8	100.0	53.2	68.8	50.1	59.7	50.2	54.5	50.3	58.8
ResNet50 [48]		55.4	89.7	90.1	96.5	50.3	66.8	56.5	67.9	50.2	55.5	50.3	68.7
Inception V3 [49]		50.1	57.1	97.8	99.7	50.0	51.9	52.9	71.6	53.1	59.4	52.8	65.2
GramNet [7]		98.9	99.9	97.6	99.8	55.7	82.1	**65.0**	**96.5**	50.0	77.9	50.3	65.7
Patch-forensics [24]		59.6	81.4	**99.4**	**100.0**	52.7	69.1	50.0	55.5	53.9	68.1	50.0	52.3
AutoGAN [32]		69.0	84.4	92.2	98.1	**67.6**	**76.3**	53.5	59.2	61.0	90.6	**65.8**	86.6
Ours		**99.9**	**100.0**	98.7	99.9	61.6	70.2	57.7	95.0	**80.9**	**98.8**	62.5	**98.9**
VGG16 [47]	**StyleGAN3**	69.8	76.2	80.8	89.0	72.3	86.4	65.0	70.6	56.5	57.5	59.5	61.7
ResNet50 [48]		59.9	72.1	65.8	83.9	81.5	89.8	61.2	70.1	54.2	58.6	58.8	63.1
Inception V3 [49]		55.0	61.5	62.1	66.3	77.5	86.6	54.4	68.9	52.0	61.2	51.8	57.7
GramNet [7]		**99.0**	**99.9**	**84.7**	**98.4**	70.0	89.7	**94.5**	**99.2**	72.5	84.7	52.0	59.3
Patch-forensics [24]		60.6	64.0	73.9	80.8	62.9	71.7	67.5	75.2	59.0	61.5	56.8	60.8
AutoGAN [32]		54.4	72.3	50.5	54.4	77.7	93.5	59.0	66.0	66.0	90.5	82.0	94.6
Ours		78.8	99.7	70.1	97.1	**88.4**	**95.5**	73.0	82.1	**79.1**	**94.2**	**79.8**	**99.7**

TABLE V
EFFECT OF DIFFERENT INPUT FEATURES OF OUR METHOD

CH	PS	FL	StyleGAN		StyleGAN2		StyleGAN3	
			ACC	AUC	ACC	AUC	ACC	AUC
✓			94.0	98.5	96.3	99.2	69.7	77.9
	✓		99.8	100.0	96.8	99.8	73.4	86.0
		✓	91.3	97.0	92.4	97.4	71.7	79.8
✓	✓		100.0	100.0	98.7	99.8	85.3	92.4
	✓	✓	100.0	100.0	96.8	99.8	78.0	94.8
✓		✓	95.9	99.2	97.0	99.6	75.3	81.5
✓	✓	✓	**100.0**	**100.0**	**98.7**	**99.8**	**88.4**	**95.5**

TABLE VI
EFFECT OF A MULTI-SCALE ENSEMBLE OF OUR METHOD

Dataset	Method	N=1		N=4	
		ACC	AUC	ACC	AUC
StyleGAN	**w/o ME**	99.9	100.0	99.9	100.0
	w/ ME	**100.0**	**100.0**	**100.0**	**100.0**
StyleGAN2	**w/o ME**	98.6	99.7	98.0	99.6
	w/ ME	**98.7**	**99.8**	**98.7**	**99.8**
StyleGAN3	**w/o ME**	87.0	95.1	82.1	92.1
	w/ ME	**87.9**	**95.5**	**85.8**	**93.2**

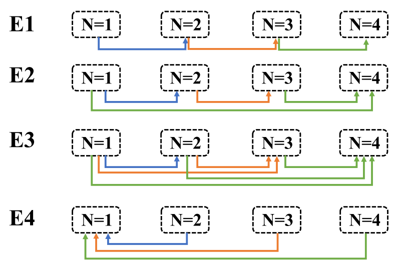

Fig. 9. Illustration of different ensemble schemes.

TABLE VII
EFFECT OF DIFFERENT MULTI-SCALE ENSEMBLE SCHEMES

Ensemble	StyleGAN		StyleGAN2		StyleGAN3	
	ACC	AUC	ACC	AUC	ACC	AUC
E1	99.8	99.9	**99.3**	**99.9**	88.7	95.4
E2	99.9	100.0	98.8	99.9	**89.9**	**96.2**
E3	99.8	100.0	98.9	99.8	89.9	95.2
E4	**100.0**	**100.0**	98.7	99.8	88.4	95.5

multi-scale setting in this experiment is not the one used in our method. It is only used to demonstrate a simple multi-scale ensemble can still improve the performance.

3) Various Ensemble Schemes: This part studies the effect of various ensemble strategies. Fig. 9 illustrates four ensemble schemes: E1 is to cascade the augmented features from each scale in order; E2 is to cascade the augmented features from each scale in order and add the global feature of $N = 1$ at the last ensemble; E3 is to cascade all augmented features from previous scales into the next scale; E4 is the ensemble scheme used in our method, which integrates the augmented features from other scales into the first scale. As shown in Table VII, all these schemes can perform well, even perfectly, on StyleGAN. For StyleGAN2 and StyleGAN3, E1 and E2 perform slightly better than others respectively. In our method, we select E4 as it achieves similar performance but can parallel train $N = 2, 3, 4$ scales and fuse with $N = 1$ finally, saving a large amount of time than other schemes.

Robustness: This part studies the robustness of our method against resizing, JPEG compression, brightness change, and additive noises, respectively.

4) Resizing: This part studies the performance of our method by testing on different image resolutions. Note that our method is independent of the image size, thus it can be applied to arbitrary sizes of images. Specifically, the images are resized to four types, 256×256, 512×512, 768×768, 1024×1024 respectively. As shown in Fig. 10 (the first column), the performance drops a lot when the resolution becomes small compared to CNN models on StyleGAN and StyleGAN2 datasets. But for the StyleGAN3 dataset, all methods drop significantly and our method achieves a similar trend to other methods. More analyses about the effect of image sizes are conducted in Section VI-E.

5) JPEG Compression: We use OpenCV to change the compression level of input images, ranging from 20 to 100.

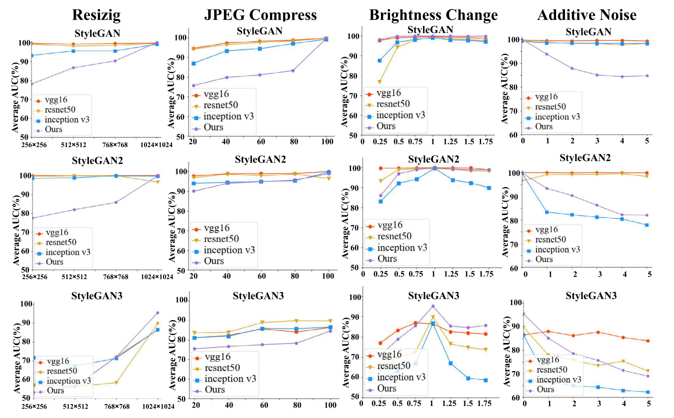

Fig. 10. Performance of different methods against resizing, JPEG compression, brightness change, and additive noise.

The larger value denotes less compression, and 100 means no compression is applied. Fig. 10 (the second column)shows the performance of our method and CNN-based methods against JPEG compression. Note these methods are trained on regular images and tested on compressed images. A similar trend is observed in these figures: all these methods drop the performance with the compression level increasing. Benefiting from a large number of parameters in CNNs, most CNN-based methods perform better than ours, *i.e.*, they drop more slowly compared to ours. Nevertheless, our method can still achieve decent performance even at the highest compression level. We leave the improvement of the robustness of our method to future work.

6) Brightness Change: Since we utilize color histograms as the feature for detection, and the color histogram is usually sensitive to light, we study the robustness of our method against brightness changes. We also use OpenCV to change the brightness of images. Specially, 1 indicates no change, and $<$ 1 indicates brighter, while $>$ 1 indicates darker. Fig. 10 (the third column) shows the curve of our method facing different brightness factors, which represents that our method can resist a certain brightness change compared to CNN models. Especially, on StyleGAN and StyleGAN3, our method performs best when the images become dark.

7) Additive Noise: For further evaluation, we add different levels of Gaussian noises to the images, making the standard deviation from 0 to 5. Note 0 means no noise is added. Fig. 10 (the last column) shows a similar curve, revealing that our method can resist certain noises on more challenging datasets compared to CNN models.

C. Results of Hybrid ForensicsForest

Table VIII shows the performance of ForensicsForest and Hybrid ForensicsForest, respectively. “FF” stands for ForensicsForest. It can be seen that after integrating the FC layer into ForensicsForest, it performs notably better on challenging StyleGAN3 datasets, improving ACC and AUC scores by 3.2% and 0.9%, respectively, demonstrating the effectiveness of added FC layers for augmented features refinement.

Ablation Study: In this part, we investigate several settings of adding FC layers into ForensicsForest, including the integration strategies and the effect of the scale of FC layers.

TABLE VIII

PERFORMANCE (%) OF FORENSICSFOREST, AND HYBRID FORENSICSFOREST, RESPECTIVELY. “FF” STANDS FOR FORENSICSFOREST

Method	StyleGAN		StyleGAN2		StyleGAN3	
	ACC	AUC	ACC	AUC	ACC	AUC
FF	**100.0**	**100.0**	98.7	99.8	88.4	95.5
Hybrid FF	99.9	100.0	**98.8**	**99.9**	**91.6**	**96.4**

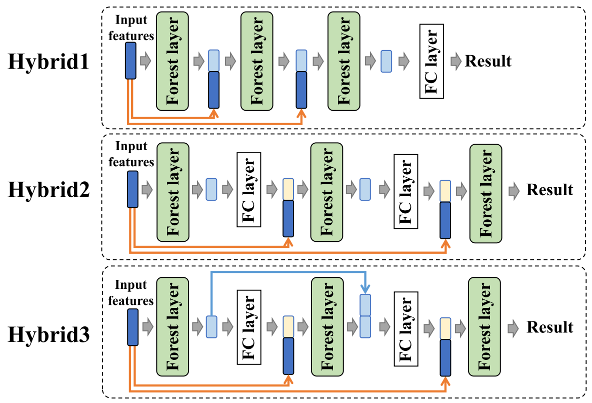

Fig. 11. Illustration of various integration strategies in Hybrid ForensicsForest.

TABLE IX

PERFORMANCE (%) OF DIFFERENT INTEGRATION STRATEGIES IN HYBRID FORENSICSFOREST

Method	StyleGAN		StyleGAN2		StyleGAN3	
	ACC	AUC	ACC	AUC	ACC	AUC
Hybrid1	99.9	100.0	98.3	99.8	85.4	94.7
Hybrid2	99.8	100.0	98.6	99.9	88.2	95.0
Hybrid3	**99.9**	**100.0**	**98.8**	**99.9**	**91.6**	**96.4**

1) Effect of Various Integration Strategies: Specifically, we study three settings: (1) we only add a FC layer at the end of our ForensicsForest (denoted as Hybrid1), (2) we simply add a FC layer after each forest layer, and send the output of FC layer into next forest layer (denoted as Hybrid2), (3) we stack the augmented features from previous forest layer to the next forest layer, as the input of FC layer in next forest layer (denoted as Hybrid3). The difference between these settings is illustrated in Fig. 11. Table IX shows the results of different settings, which reveals that Hybrid3 performs best compared to the other two settings, as it considers the knowledge from previous layers, offering sufficient information for the FC layer to precisely refine the augmented features from the forest layer. Thus we employ the third setting in our method.

2) Effect of the Scale of FC layer: This part investigates the effect of different parameters in FC layer. For better understanding, for example, we use “$a - b$” to represent adding two FC layers and these layers have a, b parameters respectively. Note that a, b are the output dimensions of FC layers and we use them to represent the scale for simplicity. Table X shows several settings of FC layers. The first row is the setting used in our method, in which we use one FC layer with 8 parameters. We also study the effect of using two FC layers with more parameters, *e.g.*, $16-8, 32-8, 64-8, 128-8$. It can be seen with the number of parameters increases, the performance is not improved as we expect. On the contrary, the first row with fewer parameters performs better than the others. We conjecture that in our integration, the efficacy of

TABLE X

Performance (%) of Hybrid ForensicsForest With Different Number of Parameters

Method	StyleGAN		StyleGAN2		StyleGAN3	
	ACC	AUC	ACC	AUC	ACC	AUC
8	99.9	**100.0**	**98.8**	**99.9**	**91.6**	**96.4**
16-8	**100.0**	100.0	98.6	99.9	88.2	95.5
32-8	100.0	100.0	98.4	99.9	89.9	96.1
64-8	100.0	100.0	98.8	99.9	89.3	95.7
128-8	100.0	100.0	98.6	99.9	87.9	96.0

TABLE XI

Performance (%) of ForensicsForest, and Divide-and-Conquer ForensicsForest Respectively. "FF" Stands for ForensicsForest. "D-and-C" Stands for Divide-and-Conquer

Method	StyleGAN		StyleGAN2		StyleGAN3	
	ACC	AUC	ACC	AUC	ACC	AUC
FF	100.0	100.0	**98.7**	99.8	**88.4**	95.5
D-and-C FF	**100.0**	**100.0**	98.6	**99.9**	87.6	**95.6**

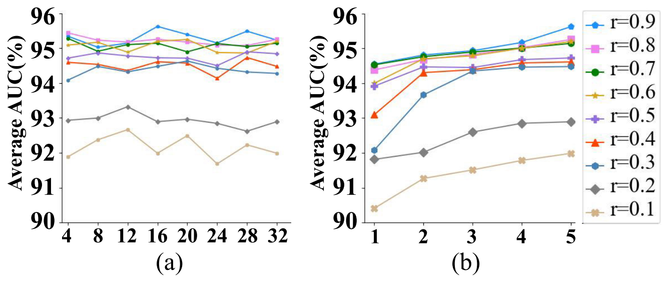

Fig. 12. (a) Effect of different r, m on Divide-and-Conquer ForensicsForest. (b) Effect of T on Divide-and-Conquer ForensicsForest. The definition of r, m, T can be found in Section V.

augmented features from the forest layer plays an important role in the final result. Despite we can employ FC layers for refinement, it may have an upper bound for the performance no matter what scale of FC layers you use.

D. Results of Divide-and-Conquer ForensicsForest

Table XI shows the performance of ForensicsForest and Divide-and-Conquer ForensicsForest, respectively. We use "D-and-C FF" to denote Divide-and-Conquer ForensicsForest. Our Divide-and-Conquer ForensicsForest can achieve competitive performance compared to our native ForensicsForest, which only slightly drops on the StyleGAN3 dataset by 0.8% on ACC, demonstrating the effectiveness of the divide-and-conquer strategy to retain the performance while saving the memory cost.

Ablation Study: We investigate the effect of Divide-and-Conquer ForensicsForest using various parameters, such as the number of forests constructed before selection, the ratio of training and validating set each time, and the number of repeated times, respectively.

1) Effect of Different Number m of Forests Before Selection and the Ratio r of Training and Validating: Set. We conduct many experiments with different m and r, and the results are illustrated in Fig. 12 (a). The x-axis is the number m of forests before the selection, and different figures represent the performance of our method using different ratios r. Note that we always select four forests from m, thus the x-axis can only start from $m = 4$, which means using all forests without selection. We study our method on eight different forest numbers m, which range from 4 to 32, and nine different ratios r, which range from 0.1 to 0.9. As shown in these figures, we can observe that the performance improves gradually with the ratio r increasing. It is consistent with our expectation, as increasing the ratio r can provide more samples for training, improving the performance subsequently. But we can also observe that the performance starts to become stable at a ratio $r = 0.4$, which demonstrates the effectiveness of our Divide-and-Conquer ForensicsForest can learn sufficient knowledge by using only a small portion of training samples. For m, our preliminary expectation is that the performance should be improved by increasing m, as larger m means providing more options for selection. However, the observation is inconsistent with our expectation, that the larger m may not bring a performance boost than less m. We conjecture that it is highly related to r, as small r means only using a small portion of training samples, that a small number of forests is sufficient to capture the full knowledge, but a large number of forests may be underfitting. With the r increasing, we can see the performance is not sensitive with m, which represents the training samples can provide sufficient knowledge to our model even though the forest number is not large.

2) Effect of Different Number of Repeated Times T: Fig. 12 (b) illustrates the performance of our method under different repeated times T with different ratio r. It can be seen that for all cases of ratio r, the trend is the same in that the performance improves with the repeated time T increasing. This is because more repeated time denotes more ForensicsForest models, which can provide more experience for final prediction.

E. Discussions

1) Why Our Method Works?: Observing that the GAN-generated faces usually exhibit distribution differences in three aspects, that are appearance, frequency, and biology respectively. This is because the objective functions used for training generative models can hardly take these aspects into account, which consequently reflects on the generated faces. For example, when training generative models, a discriminator is typically employed to distinguish if a face image is generated or not. The learning of this discriminator relies on two sets of faces, the real faces and the generated faces, which automatically learns the difference between them in a discriminator perspective. However, there are no explicit restrictions on the frequency distribution in the learning process, resulting in the inconsistency in frequency distribution between real and fake faces, Based on these aspects, we design a specific model that can effectively utilize and refine the input features to expose the more distinguishable traces.

2) Possible Limitations: Recall the robustness analysis in Section VI-B that the small size (*e.g.*, 256 × 256) may limit the performance of our method in StyleGAN compared to a larger size (*e.g.*, 1024 × 1024). To figure out the reason behind this, we resize both the training and testing images to different resolutions and employ only one feature component for evaluation. The results are visualized in Fig. 13. It can be seen that with the image size decreasing, the effectiveness

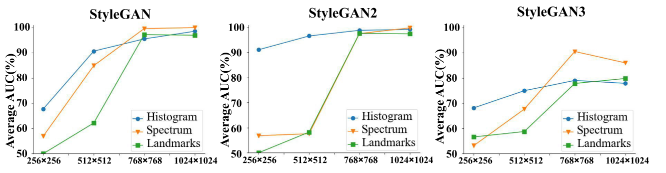

Fig. 13. Performance on images with different resolutions.

TABLE XII
EFFECT OF USING DIFFERENT INPUT FEATURES ON DIFFERENT DATASETS

CH	PS	FL	ProGAN 256 × 256		StarGAN 256 × 256		LDM 256 × 256	
			ACC	AUC	ACC	AUC	ACC	AUC
✓			82.0	92.3	100.0	100.0	80.8	88.0
	✓		96.5	99.9	100.0	100.0	100.0	100.0
		✓	93.5	98.2	99.0	99.0	63.5	68.7

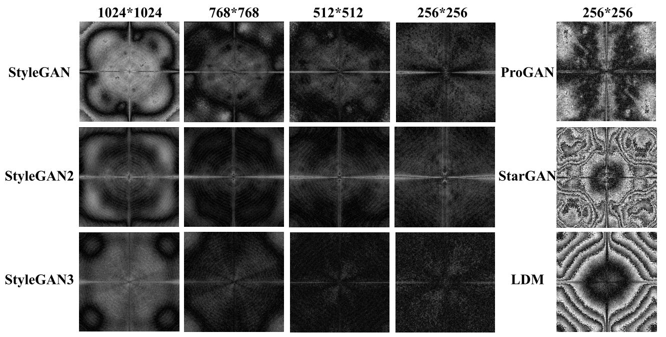

Fig. 14. Visualization of frequency difference maps of different datasets with various resolutions. The difference is multiplied by a factor of 40 for a better view.

of the frequency spectrum and landmark features are greatly disturbed compared to the color histogram, resulting in a clear performance drop. However, the results in Table III are the opposite, where our method performs very well on ProGAN, StarGAN, and LDM, even if the image size is 256 × 256. For further analysis, we show the performance of using each feature component on these datasets in Table XII. It reveals that all the features are effective even in small image sizes on ProGAN and StarGAN faces. These results are reasonable as the quality of ProGAN and StarGAN faces does not catch with StyleGAN faces, thus these features still have distinguishable abilities even in small sizes. But for LDM, which is the latest generation model, we find a similar trend in StyleGAN, that the color histogram and facial landmark become less effective, only 88.0% and 68.7% at AUC respectively. However, compared to StyleGAN, the frequency spectrum in LDM is still highly effective, which is the key to retaining favorable performance. Thus, we conjecture that the frequency spectrum in LDM is more significant than StyleGAN. For validation, we visualize the frequency difference of the faces generated by different models in Fig. 14. These maps are obtained by averaging the difference between fake and real faces. It can be seen that the frequency difference becomes inconspicuous with image size decreasing in StyleGAN datasets, while it is still apparent in ProGAN, StarGAN, and LDM. Thus, our method may not be robust when applied to the generative methods that have a frequency spectrum that is highly dependent on image sizes.

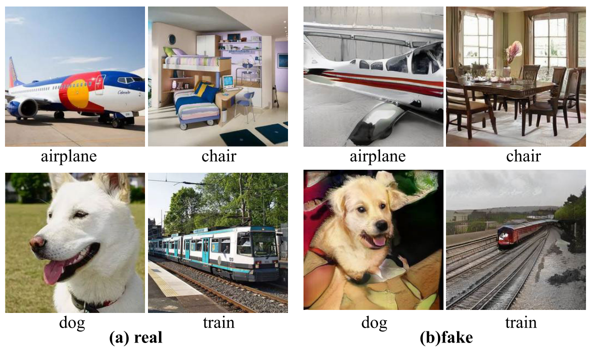

Fig. 15. Examples of real images from LSUN dataset and generated images by ProGAN.

TABLE XIII
PERFORMANCE (%) OF DIFFERENT METHODS ON PROGAN

Method	ProGAN	
	ACC	AUC
VGG16 [47]	99.7	99.9
ResNet50 [48]	99.9	100.0
Inception V3 [49]	99.9	100.0
Ours	93.7	98.5

3) Tentative Study on General Scenes: This part tentatively studies the feasibility of our method on detecting general GAN-generated images. Specifically, We select 10400 real images from LSUN dataset [63] and create the same number of images with general scenes using ProGAN [1]. This dataset covers 20 general classes and the resolution of all images is 256 × 256. Fig. 15 shows several examples. Since faces are not contained in this scenario, we only use appearance and frequency features in detection. As shown in Table XIII, our method can also achieve competitive performance.

VII. CONCLUSION

This paper describes *ForensicsForest Family*, a novel set of forest-based methods to detect GAN-generate faces. In contrast to the recent efforts of using CNNs, we investigate the feasibility of using forest models and introduce a series of multi-scale hierarchical cascade forests, which are ForensicsForest, Hybrid ForensicsForest, and Divide-and-Conquer ForensicsForest, respectively. The ForensicsForest model contains three key components: input feature extraction, hierarchical cascade forest, and multi-scale ensemble. These components are specifically designed for GAN-generated face detection. Building on the ForensicsForest, we develop the Hybrid ForensicsForest that integrates CNN layers into the model to enhance the augmented features derived from forest layers. Moreover, to reduce the memory cost of training, we develop the Divide-and-Conquer ForensicsForest, which uses a randomly selected portion of training samples to construct models and pick up the suitable components to form a final forest model. Extensive experiments are conducted on multiple types of GAN-generated faces compared to recent CNN models and dedicated CNN-based detection methods, demonstrating that our method is surprisingly effective in exposing GAN-generated faces.

REFERENCES

[1] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of GANs for improved quality, stability, and variation," in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 1–26.

[2] I. Goodfellow et al., "Generative adversarial networks," *Commun. ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[3] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of StyleGAN," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 8107–8116.

[4] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," 2013, *arXiv:1312.6114*.

[5] H. Farid, "Creating, using, misusing, and detecting deep fakes," *J. Online Trust Saf.*, vol. 1, no. 4, Sep. 2022.

[6] S.-Y. Wang, O. Wang, R. Zhang, A. Owens, and A. A. Efros, "CNN-generated images are surprisingly easy to spot for now," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 8692–8701.

[7] Z. Liu, X. Qi, and P. H. S. Torr, "Global texture enhancement for fake face detection in the wild," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 8057–8066.

[8] H. Guo, S. Hu, X. Wang, M.-C. Chang, and S. Lyu, "Robust attentive deep neural network for detecting GAN-generated faces," *IEEE Access*, vol. 10, pp. 32574–32583, 2022.

[9] A. Rössler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Niessner, "FaceForensics++: Learning to detect manipulated facial images," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1–11.

[10] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu, "Celeb-DF: A large-scale challenging dataset for DeepFake forensics," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 3204–3213.

[11] N. Carlini and H. Farid, "Evading deepfake-image detectors with white- and black-box attacks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2020, pp. 2804–2813.

[12] A. Gandhi and S. Jain, "Adversarial perturbations fool deepfake detectors," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2020, pp. 1–8.

[13] X. Yang, Y. Li, H. Qi, and S. Lyu, "Exposing GAN-synthesized faces using landmark locations," in *Proc. ACM Workshop Inf. Hiding Multimedia Secur.*, Jul. 2019, pp. 113–118.

[14] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4396–4405.

[15] T. Karras et al., "Alias-free generative adversarial networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 852–863.

[16] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo, "StarGAN: Unified generative adversarial networks for multi-domain image-to-image translation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8789–8797.

[17] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 10674–10685.

[18] J. Lu, Y. Li, J. Zhou, B. Li, and S. Lyu, "Forensics forest: Multi-scale hierarchical cascade forest for detecting GAN-generated faces," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, Jul. 2023, pp. 2309–2314.

[19] S. Prasad and K. Ramakrishnan, "On resampling detection and its application to detect image tampering," in *Proc. IEEE Int. Conf. Multimedia Expo*, Jul. 2006, pp. 1325–1328.

[20] M. Kirchner, "Fast and reliable resampling detection by spectral analysis of fixed linear predictor residue," in *Proc. 10th ACM workshop Multimedia Secur.*, Sep. 2008, pp. 11–20.

[21] B. Mahdian and S. Saic, "Blind authentication using periodic properties of interpolation," *IEEE Trans. Inf. Forensics Security*, vol. 3, no. 3, pp. 529–538, Sep. 2008.

[22] Z. Boulkenafet, J. Komulainen, and A. Hadid, "Face spoofing detection using colour texture analysis," *IEEE Trans. Inf. Forensics Security*, vol. 11, no. 8, pp. 1818–1830, Aug. 2016.

[23] N. Hulzebosch, S. Ibrahimi, and M. Worring, "Detecting CNN-generated facial images in real-world scenarios," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2020, pp. 2729–2738.

[24] L. Chai, D. Bau, S.-N. Lim, and P. Isola, "What makes fake images detectable? Understanding properties that generalize," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 103–120.

[25] W. Bai et al., "Robust texture-aware computer-generated image forensic: Benchmark and algorithm," *IEEE Trans. Image Process.*, vol. 30, pp. 8439–8453, 2021.

[26] C. Wang and W. Deng, "Representative forgery mining for fake face detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 14918–14927.

[27] H. Zhao, T. Wei, W. Zhou, W. Zhang, D. Chen, and N. Yu, "Multi-attentional deepfake detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 2185–2194.

[28] R. Han, X. Wang, N. Bai, Q. Wang, Z. Liu, and J. Xue, "FCD-Net: Learning to detect multiple types of homologous deepfake face images," *IEEE Trans. Inf. Forensics Security*, vol. 18, pp. 2653–2666, 2023.

[29] Y. Li, M.-C. Chang, and S. Lyu, "In ictu oculi: Exposing AI created fake videos by detecting eye blinking," in *Proc. IEEE Int. Workshop Inf. Forensics Security (WIFS)*, Dec. 2018, pp. 1–7.

[30] X. Yang, Y. Li, and S. Lyu, "Exposing deep fakes using inconsistent head poses," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2019, pp. 8261–8265.

[31] M. Chen, X. Liao, and M. Wu, "PulseEdit: Editing physiological signals in facial videos for privacy protection," *IEEE Trans. Inf. Forensics Security*, vol. 17, pp. 457–471, 2022.

[32] X. Zhang, S. Karaman, and S.-F. Chang, "Detecting and simulating artifacts in GAN fake images," in *Proc. IEEE Int. Workshop Inf. Forensics Security (WIFS)*, Dec. 2019, pp. 1–6.

[33] R. Durall, M. Keuper, and J. Keuper, "Watch your up-convolution: CNN based generative deep neural networks are failing to reproduce spectral distributions," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 7887–7896.

[34] Y. Li and S. Lyu, "Exposing deepfake videos by detecting face warping artifacts," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. workshops*, 2019, pp. 46–52.

[35] L. Li et al., "Face X-ray for more general face forgery detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 5000–5009.

[36] C. Tan, Y. Zhao, S. Wei, G. Gu, and Y. Wei, "Learning on gradients: Generalized artifacts representation for GAN-generated images detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 12105–12114.

[37] S. McCloskey and M. Albright, "Detecting GAN-generated imagery using color cues," 2018, *arXiv:1812.08247*.

[38] F. Matern, C. Riess, and M. Stamminger, "Exploiting visual artifacts to expose deepfakes and face manipulations," in *Proc. IEEE Winter Appl. Comput. Vis. Workshops (WACVW)*, Jan. 2019, pp. 83–92.

[39] H. Li, B. Li, S. Tan, and J. Huang, "Identification of deep network generated images using disparities in color components," *Signal Process.*, vol. 174, Sep. 2020, Art. no. 107616.

[40] R. Durall, M. Keuper, F.-J. Pfreundt, and J. Keuper, "Unmasking DeepFakes with simple features," 2019, *arXiv:1911.00686*.

[41] Y. Qian, G. Yin, L. Sheng, Z. Chen, and J. Shao, "Thinking in frequency: Face forgery detection by mining frequency-aware clues," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 86–103.

[42] J. Li, H. Xie, J. Li, Z. Wang, and Y. Zhang, "Frequency-aware discriminative feature learning supervised by single-center loss for face forgery detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 6454–6463.

[43] H. Liu et al., "Spatial-phase shallow learning: Rethinking face forgery detection in frequency domain," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 772–781.

[44] Z.-H. Zhou and J. Feng, "Deep forest: Towards an alternative to deep neural networks," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, Aug. 2017, pp. 3553–3559.

[45] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," *Handbook Systemic Autoimmune Diseases*, vol. 1, no. 4, 2009.

[46] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Jul. 1998.

[47] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.

[48] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[49] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2818–2826.

[50] A. G. Howard et al., "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*.

[51] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5987–5995.
[52] M. Tan et al., "MnasNet: Platform-aware neural architecture search for mobile," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 2815–2823.
[53] M. Tan and Q. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 6105–6114.
[54] I. Radosavovic, R. P. Kosaraju, R. Girshick, K. He, and P. Dollár, "Designing network design spaces," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 10425–10433.
[55] D. Afchar, V. Nozick, J. Yamagishi, and I. Echizen, "MesoNet: A compact facial video forgery detection network," in *Proc. IEEE Int. Workshop Inf. Forensics Security (WIFS)*, Dec. 2018, pp. 1–7.
[56] H. H. Nguyen, J. Yamagishi, and I. Echizen, "Capsule-forensics: Using capsule networks to detect forged images and videos," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2019, pp. 2307–2311.
[57] H. Dang, F. Liu, J. Stehouwer, X. Liu, and A. K. Jain, "On the detection of digital face manipulation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 5780–5789.
[58] Y. Ni, D. Meng, C. Yu, C. Quan, D. Ren, and Y. Zhao, "CORE: Consistent representation learning for face forgery detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2022, pp. 12–21.
[59] J. Cao, C. Ma, T. Yao, S. Chen, S. Ding, and X. Yang, "End-to-end reconstruction-classification learning for face forgery detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 4103–4112.
[60] Z. Yan, Y. Zhang, Y. Fan, and B. Wu, "UCF: Uncovering common features for generalizable deepfake detection," 2023, *arXiv:2304.13949*.
[61] Y. Luo, Y. Zhang, J. Yan, and W. Liu, "Generalizing face forgery detection with high-frequency features," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 16312–16321.
[62] Z. Yan, Y. Zhang, X. Yuan, S. Lyu, and B. Wu, "DeepfakeBench: A comprehensive benchmark of deepfake detection," 2023, *arXiv:2307.01426*.
[63] F. Yu, A. Seff, Y. Zhang, S. Song, T. Funkhouser, and J. Xiao, "LSUN: Construction of a large-scale image dataset using deep learning with humans in the loop," 2015, *arXiv:1506.03365*.

Jiucui Lu received the B.S. degree in computer science and technology from Qingdao University, Qingdao, China, in 2021. She is currently pursuing the M.S. degree with the Department of Computer Science and Technology, Ocean University of China, Qingdao. Her research interests include digital image forensics and adversarial attacks.

Jiaran Zhou received the B.Eng. and Ph.D. degrees in computer science and technology from Shandong University, in 2012 and 2020, respectively. She is currently a Lecturer with the Center on Artificial Intelligence, Ocean University of China. Her research interests include computer graphics, geometry processing, and artificial intelligence security.

Junyu Dong (Member, IEEE) received the B.Sc. and M.Sc. degrees in applied mathematics from the Department of Applied Mathematics, Ocean University of China, Qingdao, China, in 1993 and 1999, respectively, and the Ph.D. degree in image processing from the Department of Computer Science, Heriot-Watt University, Edinburgh, U.K., in November 2003. He is currently a Professor and the Head of the Department of Computer Science and Technology. His research interests include machine learning, big data, computer vision, and underwater image processing.

Bin Li (Senior Member, IEEE) received the B.E. degree in communication engineering and the Ph.D. degree in communication and information system from Sun Yat-sen University, Guangzhou, China, in 2004 and 2009, respectively. He was a Visiting Scholar with New Jersey Institute of Technology, Newark, NJ, USA, from 2007 to 2008. In 2009, he joined Shenzhen University, Shenzhen, China, where he is currently a Professor. He is also the Director of Guangdong Key Laboratory of Intelligent Information Processing and Shenzhen Key Laboratory of Media Security. His current research interests include multimedia forensics, image processing, and deep machine learning. He is a Senior Area Editor of IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY.

Siwei Lyu (Fellow, IEEE) received the B.S. and M.S. degrees in computer science and information science from Peking University in 1997 and 2000, respectively, and the Ph.D. degree in computer science from Dartmouth College in 2005. He is a SUNY Empire Innovation Professor with the Department of Computer Science and Engineering, University at Buffalo (State University of New York), USA. He has published over 200 refereed journals and conference papers. His research interests include digital media forensics, computer vision, and machine learning.

Yuezun Li (Member, IEEE) received the B.S. degree in software engineering from Shandong University in 2012, the M.S. degree in computer science in 2015, and the Ph.D. degree in computer science from University at Albany–SUNY, in 2020. He was a Senior Research Scientist with the Department of Computer Science and Engineering, University at Buffalo–SUNY. He is currently a Lecturer with the Center on Artificial Intelligence, Ocean University of China. His work has been published in peer-reviewed conferences and journals, including ICCV, CVPR, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, and PR. His research interests include computer vision and multimedia forensics.