2023 IEEE International Conference on Multimedia and Expo (ICME) | 978-1-6654-6891-6/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICME55011.2023.00394

Forensics Forest: Multi-scale Hierarchical Cascade Forest for Detecting GAN-generated Faces

Jiucui Lu[1], Yuezun Li[1,*], Jiaran Zhou[1], Bin Li[2], Siwei Lyu[3]
[1] Department of Computer Science and Technology, Ocean University of China, Qingdao, China
[2] Guangdong Key Laboratory of Intelligent Information Processing,
Shenzhen Key Laboratory of Media Security, Shenzhen University, Shenzhen 518060, China
[3] University at Buffalo, State University of New York, USA

***Abstract*—We describe a simple and effective method called *ForensicsForest* to detect GAN-generate faces. Instead of using the commonly used CNN models, we describe a novel multi-scale hierarchical cascade forest, which takes semantic and frequency features as input, and hierarchically cascades different levels of features for authenticity prediction. We then propose a multi-scale ensemble, which comprehensively considers different levels of information to improve the performance further. Our method is validated on state-of-the-art GAN-generated face datasets in comparison with several CNN models, which demonstrates the surprising effectiveness of our method in detecting GAN-generated faces.**

***Index Terms*—Digital forensics, GAN-generated faces 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]–[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]–[9] 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 [10], [11], 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 [12], [13]. It works because a large number of weight parameters makes the classification boundary more complicated, increasing the possibility of pushing a sample crossing the boundary with less effort. Moreover, due to the differentiability of CNNs, the attacks can be easily achieved by optimizing an objective function. Hence, resolving the above problems can make detection methods more applicable.

(a) Real (b) Fake

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

In this paper, we describe a simple and effective method called *ForensicsForest* 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 the adversarial attacks. Specifically, we propose a new architecture called Multi-scale Hierarchical Cascade Forest, which contains 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 instead of using the whole image as input, 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, which are the color histogram as semantic features and the power spectrum as frequency features. These features are concatenated and sent into a hierarchical cascade forest, which is composed of several cascade forest layers, where the features of each patch are hierarchically integrated into consecutive layers. Using this structure can iteratively process each patch, which can augment the features

* Corresponding author

1945-788X/23/$31.00 ©2023 IEEE
DOI 10.1109/ICME55011.2023.00394

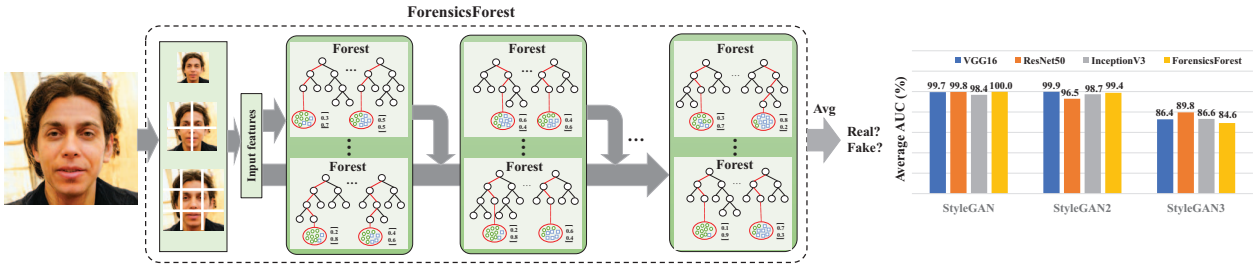

Fig. 2. Diagram of the proposed ForensicsForest for detecting GAN-generated faces. Our method can achieve competitive and even better performance compared to CNN-based detection methods.

of each patch by considering the knowledge of the previous patch while reducing the computation overload. Moreover, we propose a multi-scale ensemble, which considers different scales of features by adjusting the size of patches to fully learn the discriminative features, and ensembles these results for final prediction. Extensive experiments are conducted on the state-of-the-art GAN-generated faces (*e.g.*, StyleGAN [14], StyleGAN2 [3] and StyleGAN3 [15]), showing that our method is surprisingly effective to expose GAN-generated faces in comparison to CNN-based methods.

The contribution of this paper is summarized in two-fold:

1) Different from recent CNN-based methods, we describe a new forest-based method, a multi-scale hierarchical cascade forest, for detecting GAN-generated faces.
2) To the best of our knowledge, we are the first to investigate the feasibility of the forest model to expose GAN-generated faces, which can provide fresh insights for the following research.

II. Background and Related Works

GAN-generated Faces Detection. The early stage methods detect GAN-generated faces using statistic signals [16], [17], which becomes less effective with the improvement of GANs. The recent GAN-generated face detectors are mainly based on CNN models due to their good performance on vision tasks. The methods of [6], [7], [18] learn the detectable clues by directly training CNN models with vanilla or self-designed architectures in a supervised way. There are also many methods relying on empirically selected clues, ranging from physiological signals (*e.g.*, corneal specular highlights [9]), artifact signals (*e.g.*, upsampling artifacts [19], [20]), frequency signals (*e.g.*, [21], [22]). These clues are then sent into CNNs for final prediction. In this paper, we describe a new forest-based method called ForensicsForst to expose GAN-generated faces.

Deep Forest. Forest is the classical decision model for classification. Due to the nature of decision models, they are not differentiable. In contrast, DNN models are differentiable networks with deeper architectures, *i.e.*, multiple layers of differentiable parameterized modules. Inspired by the success of DNN models, Deep Forest [23] is proposed in a non-DNN style deep model with multiple layers. However, it usually focuses on the general classification in small scales (*e.g.*, CIFAR10 [24], MINIST [25]), which can hardly be applied to the GAN-generated faces detection. The difference between our method and Deep Forest is elaborated in Sec. III-C.

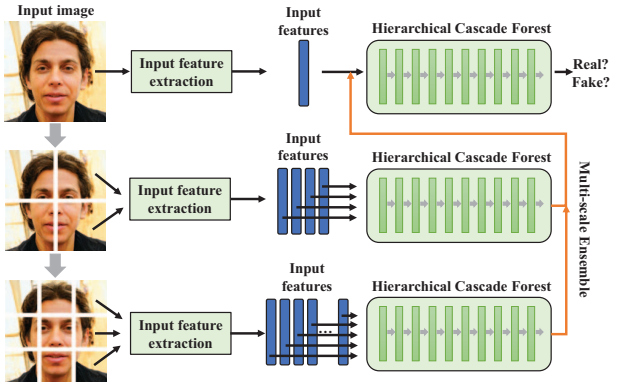

Fig. 3. Overview of the proposed multi-scale hierarchical cascade forest.

III. Method

A. ForensicsForest Structure

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

Input Feature Extraction. Instead of using the whole image as input, we use the features extracted from the image as input. 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 two types of features: semantic and frequency features.

For semantic features, we employ simple color histograms. Concretely, we first make a color histogram for each channel and sum them together as the semantic features (1×256). Moreover, we extract frequency features as complementary to semantic features. Specifically, we first convert the input image into a frequency map using Fast Fourier Transform (FFT), and then transform the frequency map into a power spectrum using the azimuthal average as the frequency feature (1×88). Fig. 4 shows the overview of input feature extraction.

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

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 by N cascade forest layers. Each layer contains two

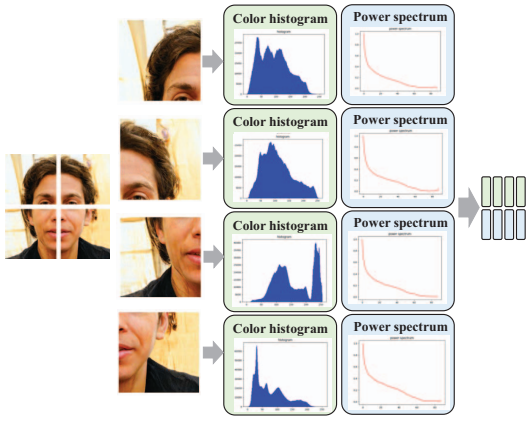

Fig. 4. Illustration of input feature extraction.

random forests and two completely random forests [26]. The random forest is composed by CART decision trees which are created using 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 features, 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 1-st layer is the input features of 1-st patch. For other blocks, the input of 1-st layer is the augmented features from the last block.

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 augmented features from this forest are concatenated with the ones from other forests ($N > 1$) as the final augmented features. By considering multiple scales, the detection performance is further improved.

B. Training and Inference

Training. Different from CNN models, our method is a decision model constructed on decision trees. Thus the training process is to create the forest structure using training images. 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

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

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

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

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. More details of training can be found in *Supplementary*.

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

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 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.

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.

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.

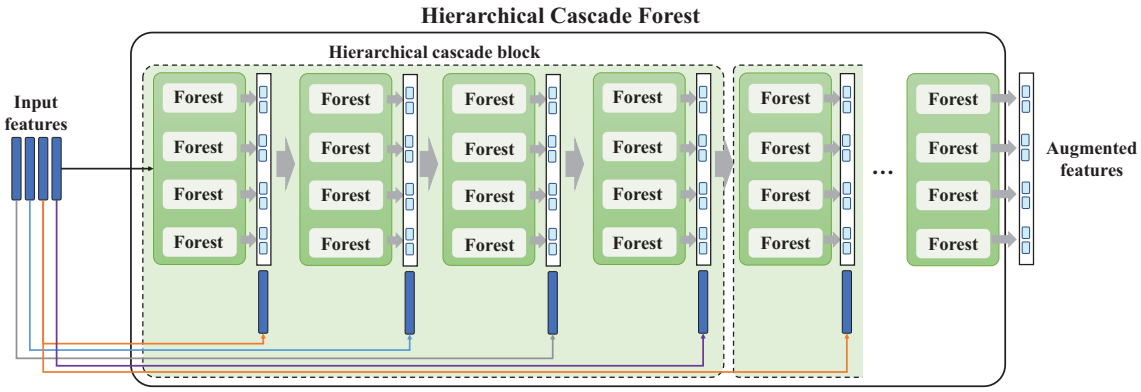

Fig. 5. Overview of the hierarchical cascade forest. The input features are hierarchically sent into the corresponding layer in a hierarchical cascade block. See text for details.

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. Experiments

A. Experimental Settings

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 dataset, we use all images of FFHQ as real set ($70,000$ images) and all images of StyleGAN faces as fake set ($100,000$ images). For StyleGAN2 dataset, we randomly select $5,000$ images from FFHQ and StyleGAN2 faces respectively. StyleGAN3 only provides 867 images. Thus we randomly select the same amount of real images from FFHQ to construct 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.

Compared CNN models. To demonstrate the efficacy of our method, we compare our method with several mainstream CNN models, which are VGG16 [27], ResNet18 [28], ResNet50, InceptionNet [29], MobileNet [30], ResNeXt [31], EfficientNet [32], MNASNet [33] and RegNet [34] respectively.

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 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. For the compared CNN models, we use their pretrained weights on ImageNet and fine-tune them on each StyleGAN dataset. The learning rate is set to 0.01. The maximum training epoch for each CNN model is set to 20. All CNN models are trained using a single Nvidia 2080ti GPU.

B. Results

Compared to CNN models. The performance of our method in comparison to CNN models is shown in Table I. The evaluation metrics are Accuracy (ACC) and Area Under Curve (AUC) following previous works [7], [9]. 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 1% in ACC on StyleGAN dataset. For example, our method surpasses ResNet50 by 2.7% in ACC, and outperforms MNASNet by a large margin, 14.5%, in ACC. It is probably because these CNNs still may not fully converge on this large dataset given a predefined epoch, due to their plenty of parameters. In contrast, our method quickly converges, thus effectively exposing the GAN-generate faces. Our method also performs very well on StyleGAN2 dataset, which achieves 96.6% in ACC and 99.4% 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 competitive performance on this dataset, which demonstrates 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	93.7	99.7	**98.9**	**100.0**	72.3	86.4
ResNet18	98.8	99.9	86.1	96.1	70.2	85.3
ResNet50	97.2	99.8	90.1	96.5	**81.5**	**89.8**
InceptionNet	86.4	98.4	87.8	98.7	77.5	86.6
MobileNet	94.5	99.3	94.6	98.8	71.7	81.6
ResNeXt	96.3	99.9	86.1	94.8	74.0	82.2
MNASNet	85.4	96.1	81.1	97.0	64.7	71.1
EfficientNet	86.3	96.6	92.1	97.7	74.1	89.3
RegNet	88.2	99.8	93.4	99.7	74.6	85.7
Ours	**99.9**	**100.0**	96.6	99.4	75.4	84.6

Table II records the time (seconds) of training corresponding methods. It can be seen that despite CNN models are trained on GPU while our method is trained on CPU, our method has significantly lower time consumption than CNN models.

Compared to GAN-generated Face Detection Methods. We also compare our method with two recent CNN-based

TABLE II
TRAINING TIME (SECONDS) OF DIFFERENT METHODS.

Method	StyleGAN	StyleGAN2	StyleGAN3
VGG16	95790.2s	3486.8s	329.5s
ResNet50	87870.4s	2640.2s	212.6s
Inception	108485.3s	3540.0s	350.6s
Ours	**14938.7s**	**458.5s**	**45.8s**

GAN-generated face detection methods, CNNDetection [6] and GramNet [7]. The results shown in Table III represent our method can achieve favorable performance on par with others. For StyleGAN3, our method can outperform GramNet but does not match CNNDetection. It is partially because CNNDetection is constructed on careful augmentation, which can learn more discriminative information than ours.

TABLE III
PERFORMANCE OF OUR METHOD IN COMPARISON TO OTHER METHODS ON THREE DATASETS.

Method	StyleGAN		StyleGAN2		StyleGAN3	
	ACC	AUC	ACC	AUC	ACC	AUC
CNNDetection	95.0	99.2	**97.3**	**99.7**	**76.8**	**95.0**
GramNet	98.9	99.9	73.2	95.6	53.2	75.4
Ours	**99.9**	**100.0**	96.6	99.4	75.4	84.6

Cross-dataset evaluation. To rule out bias in dataset, we train and test the methods on diffierent datasets. Table IV shows that our method trained StyleGAN3 can also effectively detect StyleGAN and StyleGAN2, which demonstrates that the discriminative features instead of artifacts are learned from StyleGAN3 dataset. We attribute it to the proposed hierarchical and multi-scale learning scheme.

TABLE IV
PERFORMANCE OF OUR METHOD IN COMPARISON TO OTHER METHODS WHICH ARE TRAINED ON STYLEGAN3.

Method	StyleGAN3 StyleGAN		StyleGAN3 StyleGAN2		StyleGAN3 StyleGAN3	
	ACC	AUC	ACC	AUC	ACC	AUC
VGG16	71.6	78.3	82.2	90.4	72.3	86.4
ResNet50	55.7	59.5	64.4	83.0	**81.5**	**89.8**
Inception	57.5	59.7	61.8	63.9	77.5	86.6
Ours	**85.2**	**95.9**	**93.1**	**97.4**	75.4	84.6

C. Ablation Study

Effect of Semantic and Frequency Features. Table V shows the effect of each input feature. The first and second rows denote using a color histogram (denoted as Hist) and power spectrum (denoted as Spec) respectively, and the third row combines these two features as in our method. It can be seen both of the features have the effect, and their combination has the best performance on all datasets.

TABLE V
EFFECT OF DIFFERENT INPUT FEATURES OF OUR METHOD.

Input Feature	StyleGAN		StyleGAN2		StyleGAN3	
	ACC	AUC	ACC	AUC	ACC	AUC
Hist	98.5	99.9	95.9	99.2	74.6	82.5
Spec	99.9	100.0	90.9	96.4	66.5	74.8
Hist+Spec	**99.9**	**100.0**	**96.6**	**99.4**	**75.4**	**84.6**

Effect of Multi-scale Ensemble. We compare our method (denoted as w/ ME) with a variant, which only uses one scale of input images for prediction (denoted as w/o ME). Table VI shows the performance of each case on StyleGAN dataset. 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 augmented features of the whole image for prediction, and the setting is the same for $N = 4$. 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. By using multi-scale ensemble, the performance is further improved by 1% in ACC, as it considers both global and local information, which demonstrates its effectiveness in our method.

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

	N=1		N=4	
	ACC	AUC	ACC	AUC
w/o ME	98.8	99.9	94.2	98.6
w/ ME	**99.8**	**99.9**	**99.8**	**99.9**

Various Ensemble Schemes. This part studies the effect of various ensemble strategies. Specifically, we design three ensemble schemes as E1, E2 and E3, where E1 is to cascade the augmented features from each scale in order; E2 is to cascade all augmented features from previous scales into the next scale; E3 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, E3 performs better than the other two schemes. Two observations are found in this study. The first one is cascading the augmented features from local patches into the global patch (whole image) performs better than cascading from global to local, and the second one is the simple ensemble scheme is seemingly more effective than the complex one, *e.g.*, E1 outperforms E2 on StyleGAN3 dataset by a large margin.

TABLE VII
EFFECT OF DIFFERENT MULTI-SCALE ENSEMBLE SCHEMES.

Ensemble	StyleGAN		StyleGAN2		StyleGAN3	
	ACC	AUC	ACC	AUC	ACC	AUC
E1	99.8	99.9	92.3	98.2	70.2	82.3
E2	99.9	99.9	94.5	98.7	67.9	76.8
E3	**99.9**	**100.0**	**96.6**	**99.4**	**75.4**	**84.6**

Various Predictors. The output of the hierarchical cascade forest can be sent into another predictor for further refinement. Specifically, we study four types of predictors, which are None, Forest, LightGBM, and XGBoost respectively. None denotes directly averaging the output as the prediction. Forest denotes using another random forest for the final prediction. LightGBM and XGBoost are also constructed on the forest but have different ways to grow trees. Table VIII shows that LightGBM and XGBoost can improve performance. In particular, XGBoost performs the best on the more challenging StyleGAN3 dataset. Thus we utilize XGBoost as the predictor in our method.

Robustness. This part studies the robustness of our method against JPEG compression and brightness change. Specifically, we use OpenCV to change the compression level of input images, ranging from 20 to 100. The larger value denotes less compression. 100 means no compression is applied. For brightness change, 1 indicates no change, < 1 indicates

TABLE VIII
EFFECT OF DIFFERENT PREDICTORS.

Predictor	StyleGAN		StyleGAN2		StyleGAN3	
	ACC	AUC	ACC	AUC	ACC	AUC
None	99.8	99.9	95.8	99.1	67.3	70.7
Forest	99.8	99.9	95.8	98.9	67.6	72.7
LightGBM	99.9	100.0	**96.7**	99.4	**76.6**	80.1
XGBoost	**99.9**	**100.0**	96.6	**99.4**	75.4	**84.6**

brighter, and > 1 indicates darker. Fig. 6 shows the performance of different methods. Note these methods are trained on regular images and tested on processed images. A similar trend shows that our method still achieves decent performance even at the highest compression level. Benefiting from the large number of parameters of CNNs, most of the CNN-based methods perform better than ours. We leave the improvement of robustness to future work. Moreover, it can be seen our method can resist a certain brightness change.

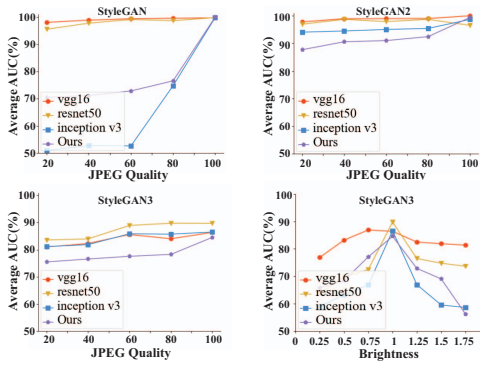

Fig. 6. Performance of different methods against JPEG compression and brightness change.

V. CONCLUSION

This paper describes a new forest-based method to detect GAN-generate faces. In contrast to the recent efforts of using CNNs, we propose a new multi-scale hierarchical cascade forest. Our method contains three main components of input feature extraction, hierarchical cascade forest, and multi-scale ensemble respectively. Extensive experiments are conducted on multiple types of GAN-generated faces in comparison with recent CNN models, demonstrating that our method is surprisingly effective in exposing GAN-generated faces.

Acknowledgement. This work is supported by the Fundamental Research Funds for the Central Universities and China Postdoctoral Science Foundation under Grant No. 2021TQ0314 and 2021M703036.

REFERENCES

[1] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen, "Progressive growing of GANs for improved quality, stability, and variation," in *ICLR*, 2018.
[2] Ian Goodfellow et al., "Generative adversarial networks," *Communications of the ACM*, 2020.
[3] Tero Karras, Samuli Laine, et al., "Analyzing and improving the image quality of stylegan," in *CVPR*, 2020.
[4] Diederik P Kingma and Max Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.
[5] Hany Farid, "Creating, using, misusing, and detecting deep fakes," *Journal of Online Trust and Safety*, 2022.
[6] Sheng-Yu Wang, Oliver Wang, et al., "Cnn-generated images are surprisingly easy to spot... for now," in *CVPR*, 2020.
[7] Zhengzhe Liu, Xiaojuan Qi, and Philip HS Torr, "Global texture enhancement for fake face detection in the wild," in *CVPR*, 2020.
[8] Chenqi Kong, Baoliang Chen, Wenhan Yang, Haoliang Li, Peilin Chen, and Shiqi Wang, "Appearance matters, so does audio: Revealing the hidden face via cross-modality transfer," *IEEE TCSVT*, 2021.
[9] Hui Guo, Shu Hu, Xin Wang, Ming-Ching Chang, and Siwei Lyu, "Robust attentive deep neural network for detecting gan-generated faces," *IEEE Access*, 2022.
[10] Andreas Rössler, Davide Cozzolino, et al., "Faceforensics++: Learning to Detect Manipulated Facial Images.," in *ICCV*, 2019.
[11] Yuezun Li, Xin Yang, Pu Sun, Honggang Qi, and Siwei Lyu, "Celeb-df: A large-scale challenging dataset for deepfake forensics," in *CVPR*, 2020.
[12] Nicholas Carlini and Hany Farid, "Evading deepfake-image detectors with white-and black-box attacks," in *CVPR*, 2020.
[13] Apurva Gandhi and Shomik Jain, "Adversarial perturbations fool deepfake detectors," in *IJCNN*, 2020.
[14] Tero Karras, Samuli Laine, and Timo Aila, "A style-based generator architecture for generative adversarial networks," in *CVPR*, 2019.
[15] Tero Karras, Miika Aittala, et al., "Alias-free generative adversarial networks," in *NeurIPS*, 2021.
[16] Xin Yang, Yuezun Li, and Siwei Lyu, "Exposing deep fakes using inconsistent head poses," in *ICASSP*, 2019.
[17] Haodong Li, Bin Li, Shunquan Tan, and Jiwu Huang, "Identification of deep network generated images using disparities in color components," *Signal Processing*, 2020.
[18] Nils Hulzebosch, Sarah Ibrahimi, and Marcel Worring, "Detecting cnn-generated facial images in real-world scenarios," in *CVPR workshop*, 2020.
[19] Xu Zhang, Svebor Karaman, and Shih-Fu Chang, "Detecting and simulating artifacts in gan fake images," in *WIFS*, 2019.
[20] Ricard Durall, Margret Keuper, and Janis Keuper, "Watch your up-convolution: Cnn based generative deep neural networks are failing to reproduce spectral distributions," in *CVPR*, 2020.
[21] Jiaming Li, Hongtao Xie, et al., "Frequency-Aware Discriminative Feature Learning Supervised by Single-Center Loss for Face Forgery Detection.," in *CVPR*, 2021.
[22] Honggu Liu, Xiaodan Li, et al., "Spatial-Phase Shallow Learning: Rethinking Face Forgery Detection in Frequency Domain.," in *CVPR*, 2021.
[23] Zhi-Hua Zhou and Ji Feng, "Deep forest: Towards an alternative to deep neural networks.," in *IJCAI*, 2017.
[24] Alex Krizhevsky, Geoffrey Hinton, et al., "Learning multiple layers of features from tiny images," 2009.
[25] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, 1998.
[26] Leo Breiman, "Random forests," *Machine learning*, 2001.
[27] Karen Simonyan and Andrew Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.
[28] Kaiming He, Xiangyu Zhang, et al., "Deep residual learning for image recognition," in *CVPR*, 2016.
[29] Christian Szegedy, Vincent Vanhoucke, et al., "Rethinking the inception architecture for computer vision," in *CVPR*, 2016.
[30] Andrew G Howard, Menglong Zhu, et al., "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.
[31] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He, "Aggregated residual transformations for deep neural networks," in *CVPR*, 2017.
[32] Mingxing Tan and Quoc Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *ICML*, 2019.
[33] Mingxing Tan, Bo Chen, et al., "Mnasnet: Platform-aware neural architecture search for mobile," in *CVPR*, 2019.
[34] Ilija Radosavovic, Raj Prateek Kosaraju, et al., "Designing network design spaces," in *CVPR*, 2020.