Unmasking DeepFakes with simple Features

Ricard Durall[1,2,3] Margret Keuper[4] Franz-Josef Pfreundt[1] Janis Keuper[1,5]
[1]Fraunhofer ITWM, Germany
[2]IWR, University of Heidelberg, Germany
[3]Fraunhofer Center Machine Learning, Germany
[4]Data and Web Science Group, University Mannheim, Germany
[5]Institute for Machine Learning and Analytics, Offenburg University, Germany

***Abstract*—Deep generative models have recently achieved impressive results for many real-world applications, successfully generating high-resolution and diverse samples from complex data sets. Due to this improvement, fake digital contents have proliferated growing concern and spreading distrust in image content, leading to an urgent need for automated ways to detect these AI-generated fake images.**

Despite the fact that many face editing algorithms seem to produce realistic human faces, upon closer examination, they do exhibit artifacts in certain domains which are often hidden to the naked eye. In this work, we present a simple way to detect such fake face images - so-called *DeepFakes*. Our method is based on a classical frequency domain analysis followed by a basic classifier. Compared to previous systems, which need to be fed with large amounts of labeled data, our approach showed very good results using only a few annotated training samples and even achieved good accuracies in fully unsupervised scenarios. For the evaluation on high resolution face images, we combined several public data sets of real and fake faces into a new benchmark: *Faces-HQ*. Given such high-resolution images, our approach reaches a perfect classification accuracy of 100% when it is trained on as little as 20 annotated samples. In a second experiment, in the evaluation of the medium-resolution images of the *CelebA* data set, our method achieves 100% accuracy supervised and 96% in an unsupervised setting. Finally, evaluating a low-resolution video sequences of the *FaceForensics*++ data set, our method achieves 91% accuracy detecting manipulated videos.

Source Code: https://github.com/cc-hpc-itwm/DeepFakeDetection

***Index Terms*—GAN images, DeepFake, Image forensic, Forgery detection**

I. INTRODUCTION

Over the last years, the increasing sophistication of smartphones and the growth of social networks have led to a gigantic amount of new digital object contents. This tremendous use of digital images has been followed by a rise of techniques to alter image contents. Until recently, such techniques were beyond the reach of most users since they were dull and time-consuming and they required a high domain expertise on computer vision. Nevertheless, thanks to the recent advances of machine learning and the accessibility to large-volume training data, those limitations have gradually faded away. As a consequence, the time for fabrication and manipulation of digital contents has significantly decreased, allowing even amateur users the modification of contents at their will.

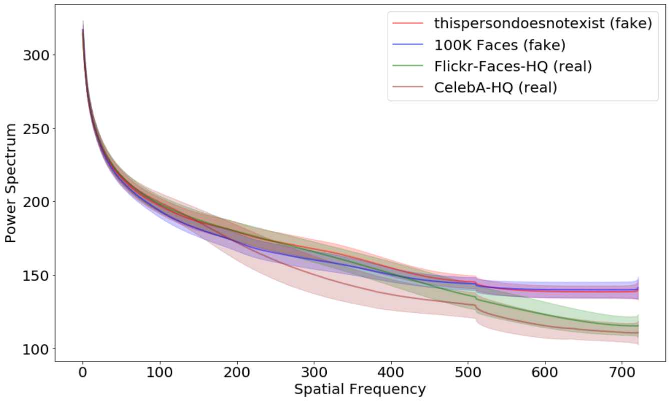


Fig. 1: 1D power spectrum statistics from each sub-data set from Faces-HQ. The higher the frequency, the bigger is the difference between real or fake data.

In particular, deep generative models have lately been extensively used to produce artificial images with realistic appearance. Theses models are based on deep neural networks which are able to approximate the true data distribution of a given training set. Hence, one can sample from the learned distribution and add variations. Two of the most commonly used and efficient approaches are Variational Autoencoders (VAE) [16] and Generative Adversarial Networks (GAN) [11]. Especially GAN approaches have lately been pushing the limits of state-of-the-art results, improving the resolution and quality of images produced [4], [14], [15]. As a result, deep generative models are opening the door to a new vein of AI-based fake image generation leading to a fast dissemination of high quality tampered image content. While significant developments have been made for image forgery detection, it still remains a hard task since most current methods rely on deep learning approaches, which require large amounts of labeled training data.

In this paper, we address the problem of detecting these artificial image contents, more specifically, fake faces. In order to determine the nature of these pictures, we introduce a new machine learning based method. Our approach relies on a classical frequency analysis of the images that reveals different behaviors at high frequencies. Fig. 1 shows how different a certain range of frequency components behave when the

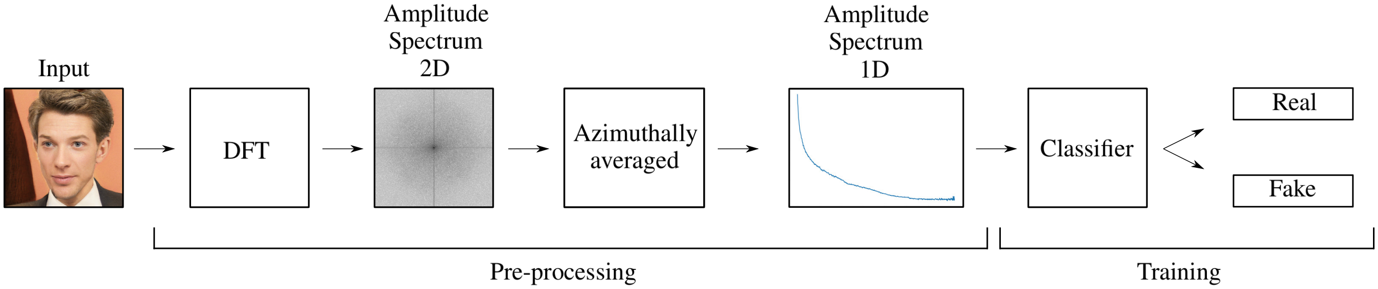


Fig. 2: Overview of the processing pipeline of our approach. It contains two main blocks, a feature extraction block using DFT and a training block, where a classifier uses the new transformed features to determine whether the face is real or not. Notice that input images are transformed to grey-scale before DFT.

images have been artificially generated.

Our method detects such artifacts by analyzing the frequency domain followed by a simple supervised or unsupervised classifier. Notice that this suggested pipeline does not involve nor requires vast quantities of data, which is a very convenient property for those scenarios that suffer from data scarcity. In addition, we introduce a new data set *Faces-HQ*, which we used to complement the *CelebA* data set and *FaceForensics++* data set [25], for our experimental evaluation.

Overall, our contributions are summarized as follows:

- We introduce a novel classification pipeline for artificial face detection based on a frequency domain analysis.
- We provide a public data set (*Faces-HQ*) of high quality images containing real and fake faces from a set of different public databases.
- We demonstrate how we successfully learn to detect forgery: extensive experiments on high and medium-resolution images of the *Faces-HQ* and *CelebA* data sets showed 100% accuracy. Additionally, the evaluation of the *FaceForensics++* data set with low-resolution videos reached 91% accuracy.

II. Related Work

In this section, we briefly review the related seminal work on high-resolution artificial images forensics and deepfake images, as well as forgery detection. In particular, we focus our attention on spotting tampered images generated by GAN-based methods.

Traditional image forensics methods can be classified according to the image features that they target, such as local noise estimation [23], pattern analysis [10], illumination modeling [9] and steganalysis feature classification [7]. However, with the deep learning breakthrough, the computer vision community has radically steered towards neural networks techniques. For example, [8], [28] are recent works based on Convolutional Neural Networks (CNN). These CNN-based approaches also aim to capture the aforementioned image features, but in an inexplicit way.

In 2014, *Goodfellow et al.* introduced an adversarial framework (GAN) which marked a milestone in generative models. In particular, the image generation has been improved significantly, leading to a striking progress on artificial faces [5] among others. As a consequence, new image and video manipulation techniques known as DeepFake have emerged and established themselves online over the last few months. This occurrence of events on digital image forensics has been drawing an ever increasing attention trying to detect GAN generated images or videos.

The lack of eye blinking [18] is one drawback observed, when the videos are artificially created. This is due to the scarcity of training images including photographs with the subject's eyes closed. Nevertheless, this detection can be circumvented by adding images with closed eyes in training. Finding unnatural head poses, is also an extend technique [26], in order to detect tampered digital contents. On the other hand, the works [17], [22] analyze the color-space features from GAN generated images and real images, and use the disparity to classify them.

Other approaches [2], [21], [27], rather than leveraging explicit lacks or failures, rely on CNNs to distinguish GAN's output from real images. In the same vein, [13] introduces deep forgery discriminator with a contrastive loss function and [12] incorporates temporal domain information by employing Recurrent Neural Networks (RNNs) upon CNNs. While deep learning methods show promising performance, a key concern is that all these methods can be easily learnt by the GAN. In particular, by incorporating them in to the GAN's discriminator, the generator can be fine-tuned to learn a countermeasure for any differentiable forensic.

III. Method

In the following section, we describe our approach in detail. Figure 2) gives an overview of the processing steps.

A. Frequency Domain Analysis

Frequency domain analysis is of utmost importance in signal processing theory and applications. In particular in the computer vision domain, the repetitive nature or the frequency

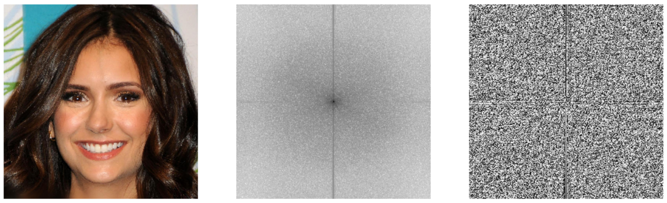


Fig. 3: Example of a DFT applied to a sample. (Left) Input image [1]. (Center) Power Spectrum. (Right) Phase Spectrum.

characteristics of images can be analyzed on a space defined by Fourier transform. Such transformation consists in a spectral decomposition of the input data indicating how the signal's energy is distributed over a range of frequencies. Methods based on frequency domain analysis have shown wide applications in image processing, such as image analysis, image filtering, image reconstruction and image compression.

1) Discrete Fourier Transform: The Discrete Fourier Transform (DFT) is a mathematical technique to decompose a discrete signal into sinusoidal components of various frequencies ranging from 0 (i.e., constant frequency, corresponding to the image mean value) up to the maximum representable frequency, given the spatial resolution. It is the discrete analogon of the continuous Fourier Transform for signals sampled on equidistant points. For 2-dimensional data of size $M \times N$, it can be computed as

$$X_{k,\ell} = \sum_{n=0}^{N-1} \sum_{0}^{M-1} x_{n,m} \cdot e^{-\frac{i2\pi}{N}kn} \cdot e^{-\frac{i2\pi}{M}\ell m}. \tag{1}$$

The frequency-domain representation of a signal (X_k) carries information about the signal's amplitude and phase at each frequency. Fig. 3 depicts the complex output information (power and phase). Notice that the amplitude spectrum is the square root power spectrum.

2) Azimuthal Average: After applying a Fourier Transform to a sample image, the information is represented in a new domain but within the same dimensionality. Therefore, given that we work with images, the output still contains 2D information. We apply azimuthal averaging to compute a robust 1D representation of the FFT power spectrum. It can be seen as a compression, gathering and averaging similar frequency components into a vector of features. In this way, we can reduce the amount of features without losing relevant information. Furthermore, throughout this compression, we achieve a more robust representation of the input. Fig. 4 shows a visual example of such method.

B. Classifier Algorithms

Classification is the task to learn a general mapping from the attribute space to descrete classes, using specific examples of instances, each represented by a vector of attribute values and their acording lable.

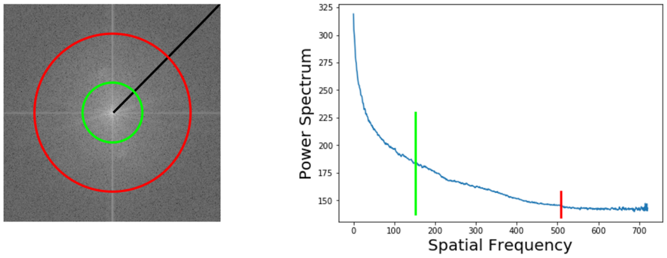


Fig. 4: Example of an azimuthal average. (Left) Power Spectrum 2D. (Right) Power Spectrum 1D. Each frequency component is the radial average from the 2D spectrum.

1) Logistic Regression: One of the technically simplest (linear) classification algorithms is the Logistic Regression (LR). It is a simple statistical model that employs a logistic function (see Formula (4)) to model a binary dependent variable. The output from the hypothesis h is the estimated probability. This is used to infer how confident predicted value can be given an input $\mathbf{x}$. Logistic regression is formulated as

$$h_{\mathbf{w}}(\mathbf{x}) = \frac{1}{1 + e^{-\mathbf{w}^T\mathbf{x}}} \tag{2}$$

The underlying algorithm of maximum likelihood estimation determines the regression coefficient $\mathbf{w}$ for the model that accurately predicts and fits the probability of the binary dependent variable. The algorithm stops when the convergence criterion is met or the maximum number of iterations is reached.

2) Support Vector Machines: Support Vector Machines (SVMs) [3], [6] are among the most widely used learning algorithms for (non-linear) data classification. The target of the SVM formulation is to produce a model (based on the training data) which will identify an optimal separating hyperplane, maximizing the margin between different classes. Given a training set of instance-label pairs (x_i, y_i), $i = 1, ..., l$ where $x_i \in R^n$ and $\mathbf{y} \in \{1, -1\}^l$, Training of SVMs is implemented by the solution of the following optimization problem

$$\begin{aligned} \min_{\mathbf{w},b,\boldsymbol{\xi}} \quad & \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{l}\xi_i \\ \text{s.t.} \quad & y_i(\mathbf{w}^T\phi(x_i) + b) \geq 1 - \xi_i, \\ & \xi_i \geq 0, \end{aligned} \tag{3}$$

where $\mathbf{w}$ and b are the parameters of our classifier, ξ is the slack variable and $C > 0$ the penalty parameter of the error term.

Here training vectors x_i are mapped into a higher dimensional space by the function ϕ. The training objective of SVMs is to find a linear separating hyperplane with the

[1] Notice that we convert the input image to gray-scale before applying FFT.

maximal margin in this higher dimensional space.

3) K-Means Clustering: While supervised classification algorithms like SVM and LR rely on labeled training example to learn a classification, we also want to test the detection performance in the absence of any labeled data. Clustering is an unsupervised machine learning technique which finds similarities in the data points and group similar data points together. The key assumption is that nearby points in the feature space exhibit similar qualities and they can be clustered together. Clustering can be done using different techniques like K-means clustering.

The K-means objective function is defined as

$$J = \sum_{k=1}^{K} \sum_{i=1}^{m} ||x_i - \mu_k||^2 \tag{4}$$

where K and m are the number of clusters and samples respectively.
A common approach to heuristically approximate solutions is to iteratively identify nearby features based on the distances calculated from initial centroids μ. Then, these features are assigned to the closest cluster and the centroids are re-estimated. Since the amount of clusters is determined by the user, it can be easily employed in classification where we divide data into K clusters with K equal to or greater than the number of classes.

IV. Experiments

In this section, we show results for a series of experiments evaluating the effectiveness of our approach. First, we introduce a new high-resolution data set, called *Faces-HQ*, together with its training settings and experiments, and we discuss our results in detail. In order to verify our approach, we also evaluate on the *CelebA* data set [19], which contains medium-resolution images, and on the *FaceForensics++* data set [25], which contains low-resolution video sequences.

A. Faces-HQ

1) Data set: to the best of our knowledge, currently no public data set is providing high resolution images with annotated fake and real faces. Therefore, we have created our own data set from established sources, called *Faces-HQ*[2]. In order to have a sufficient variety of faces, we have chosen to download and label the images available from the *CelebA-HQ* data set [14], *Flickr-Faces-HQ* data set [15], 100K Faces project [1] and *www.thispersondoesnotexist.com*. In total, we have collected 40K high quality images, half of them real and the other half fake faces. Table I contains a summary.

	# of samples	category	label
CelebA-HQ data set [14]	10000	Fake	0
Flickr-Faces-HQ data set [15]	10000	Fake	0
100K Faces project [1]	10000	Real	1
www.thispersondoesnotexist.com	10000	Real	1

TABLE I: *Faces-HQ* data set structure.

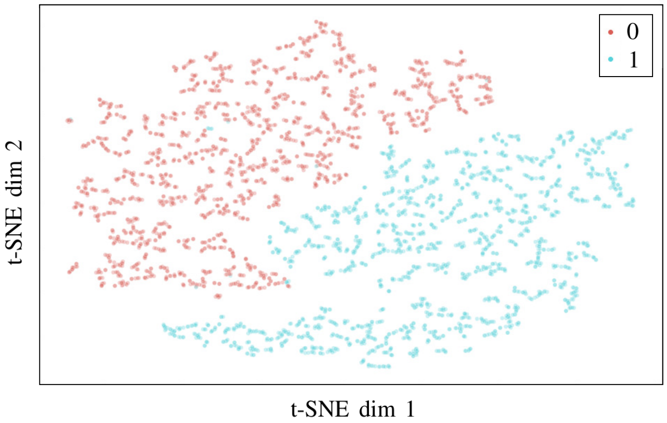


Fig. 5: T-SNE visualization of 1D Power Spectrum on a random subset from *Faces-HQ* data set. We used a perplexity of 4 and 4000 iterations to produce the plot.

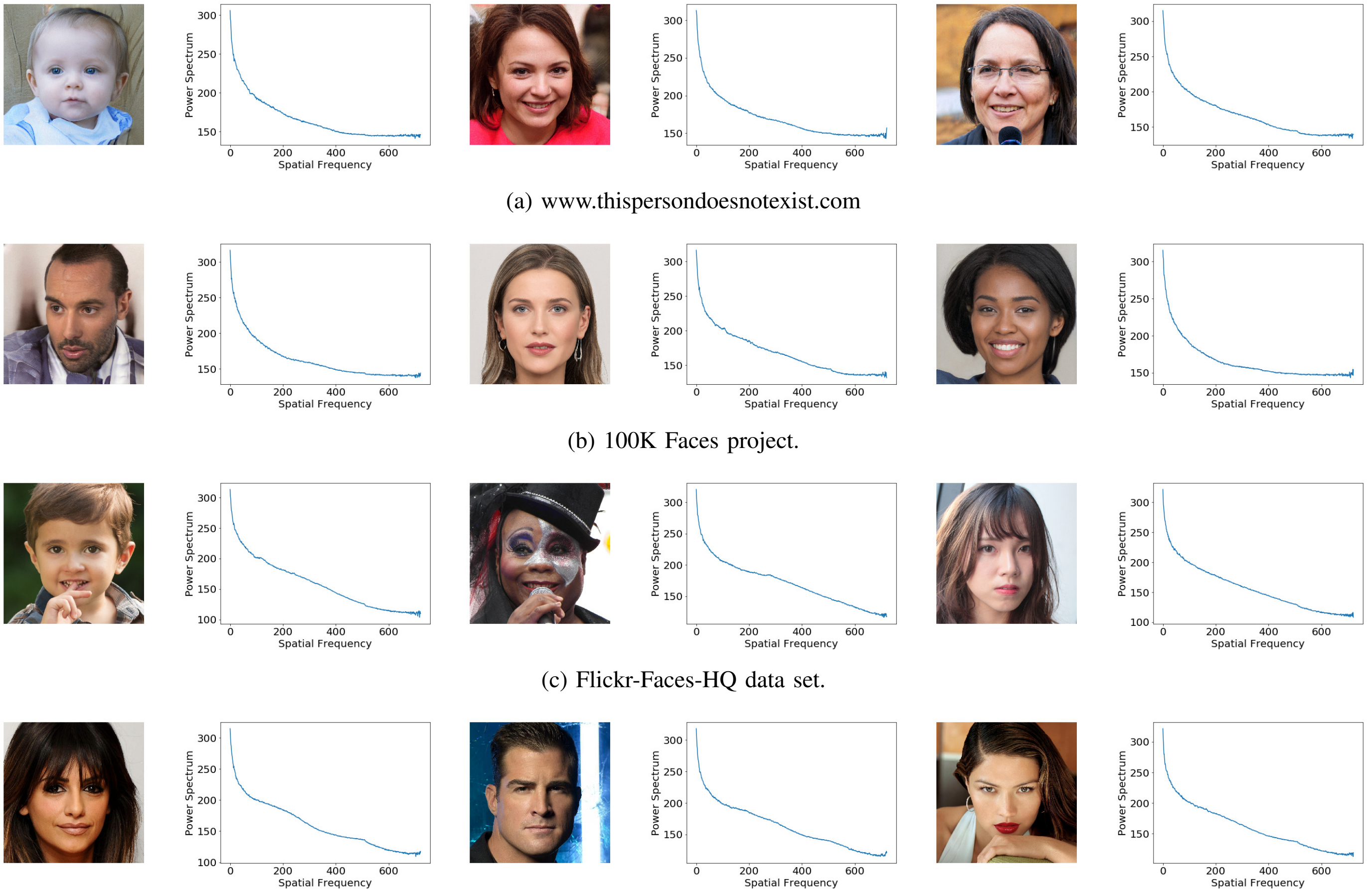
2) Training Setting: as shown in Fig. 2, our pipeline is split into two parts. On the one hand, at pre-processing time, we take the whole data set and we transform every sample from the spatial domain to the 1D frequency domain, reducing 1024x1024x3 high quality color images to 722 features (1D Power Spectrum). This method is formed by a Discrete Fourier Transform followed by an azimuthally average. The transformation can be substantially optimized by employing the Fast Fourier Transform. Notice that after applying the transformation, we use only the power spectrum since it already contains enough information for the classifier. A first visualization (see Fig. 5) using t-Distributed Stochastic Neighbor Embedding [20] (t-SNE) reveals a clear clustering of fake and real samples in this feature space.

On the other hand, once the pre-processing step is finished, we start training the classifier engine. First of all, we divide the transformed data into training and testing sets, with 20% for the testing stage and use the remaining 80% as the training set. Then, we train a classifier with the training data and finally evaluate the accuracy on the testing set. Our goal is to distinguish, real and fake faces, thus we need to use a binary classifier.

3) Method 1D Power Spectrum: looking at Fig. 6, one can observe that there is a certain repetitive behavior or pattern on the 1D Power Spectrum on those images that belong to the same class. Just by checking individual samples, it is possible to conclude that real and fake images behave in noticeable different spectra at high frequencies, and therefore they can be easily classified.

Driven by this phenomenon, we have evaluated a significant

[2] Faces-HQ data has a size of 19GB. Download: https://cutt.ly/6enDLYG



(a) www.thispersondoesnotexist.com

(b) 100K Faces project.

(c) Flickr-Faces-HQ data set.

(d) CelebA-HQ data set.

Fig. 6: Samples from the different data sets gathered on *Faces-HQ* data set. It is possible to observe on the 1D Power Spectrum some similitudes between images belonging to the same class and differences otherwise. For instance, real faces (c) and (d) do not have flat regions at high frequencies, whereas fake (a) and (b) have them.

	80% (train) - 20% (test)		
# samples	SVM	Logistic Reg.	K-Means
4000	100%	100%	77%
1000	100%	100%	77%
100	100%	100%	75%
20	100%	100%	72%

TABLE II: Test accuracy using SVM, logistic regression and k-means classifier under different data settings.

subset of images (4000 in total, 1000 of each sub-data set) and we have computed basic statistics to try to find a more general representation that help to simplify the problem. Fig. 1 plots the mean and the standard deviation of each sub-data set and corroborates the observable and distinguishable trend that real and fake images have. Motivated by this observations we have carried out a set of tests to determine the extent to which our approach successfully detects deepfakes and how much data is needed to train the model. In our experiments, we have implemented one classifier based on support vector machines (SVMs) with a radial basis function kernel and one based on logistic regression. We have run an initial experiment using 80% of the data for training and 20% for testing. We have utilized this configuration for different amount of samples (4000, 1000, 100, 20) equally distributed (see Table II).

to from	100	200	300	400	500	600	722
0	58%	69%	85%	89%	98%	**100%**	**100%**
100	-	72%	86%	89%	98%	**100%**	**100%**
200	-	-	85%	87%	99%	**100%**	**100%**
300	-	-	-	84%	98%	**100%**	**100%**
400	-	-	-	-	93%	**100%**	**100%**
500	-	-	-	-	-	**100%**	**100%**
600	-	-	-	-	-	-	**100%**

TABLE III: Test accuracy using SVM classifier.

After testing the effectiveness and efficiency of our transformed features, we have conducted a another round of experiments to determine the impact of different frequency components. Given the 722 features from 1D Power Spectrum, we have analyzed the relevance of different frequencies by grouping them into 28 sub-sections. Table III,Table IV and Table V show the accuracy results on SVM, logistic regression and K-means respectively. The rows indicate where the chunk

to from	100	200	300	400	500	600	722
0	58%	70%	86%	90%	98%	**100%**	**100%**
100	-	72%	88%	90%	98%	**100%**	**100%**
200	-	-	86%	89%	99%	**100%**	**100%**
300	-	-	-	85%	98%	**100%**	**100%**
400	-	-	-	-	92%	**100%**	**100%**
500	-	-	-	-	-	**100%**	**100%**
600	-	-	-	-	-	-	99%

TABLE IV: Test accuracy using logistic regression classifier.

to from	100	200	300	400	500	600	722
0	75%	75%	75%	72%	74%	74%	77%
100	-	75%	75%	74%	74%	74%	78%
200	-	-	73%	72%	74%	74%	75%
300	-	-	-	72%	74%	76%	76%
400	-	-	-	-	75%	77%	74%
500	-	-	-	-	-	**80%**	77%
600	-	-	-	-	-	-	74%

TABLE V: Test accuracy using k-means classifier.

of frequencies starts, and the column where it ends. For example, there is a chunk with 0.86 accuracy that contains frequencies from 100 to 300.

B. *CelebA*

1) Data set: CelebFaces Attributes (*CelebA*) data set [19] consists of 202,599 celebrity face images with 40 variations in facial attributes. The dimensions of the face images are 178x218x3, which can be considered to be a medium resolution in our context.

2) Training Setting: In order to train our forgery detection classifier we need both real and fake images. We used the real images from the *CelebA* data set. On the same set, we then train a DCGAN [24] to generate realistic but fake images. We split the data set into 162,770 images for training and 39,829 for testing, and we crop and resize the initial 178x218x3 size images to 128x128x3. Once the model is trained, we can conduct the classification experiments on medium-resolution scale.

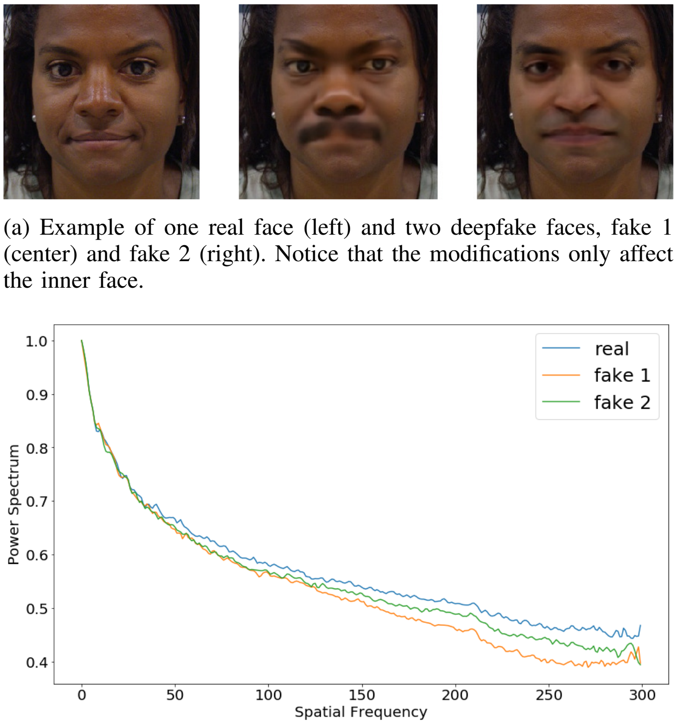
3) Results: We follow the same procedure as in the previous experiments. Table VI) shows perfect classification accuracy in the supervised, and also very good results in unsupervised clustering.

	80% (train) - 20% (test)		
# samples	SVM	Logistic Reg.	K-Means
2000	100%	100%	96%

TABLE VI: Test accuracy using SVM, logistic regression and k-means classifier.

C. *FaceForensics++*

1) Data set: *FaceForensics++* [25] is a forensics data set consisting of video sequences that have been modified with different automated face manipulation methods. Additionally, it is hosting DeepFakeDetection Data set. In particular, this data set contains 363 original sequences from 28 paid actors in 16 different scenes as well as over 3000 manipulated videos using DeepFakes and their corresponding binary masks. All videos contain a trackable, mostly frontal face without occlusions which enables automated tampering methods to generate realistic forgeries.



(a) Example of one real face (left) and two deepfake faces, fake 1 (center) and fake 2 (right). Notice that the modifications only affect the inner face.

(b) Normalized and interpolated 1D Power Spectrum from the previous images.

Fig. 7: Cropped samples from DeepFakeDetection data set and their corresponding statistics.

2) Training Setting: The employed pipeline for this data set is the same as for *Faces-HQ* data set and *CelebA*, but with an additional block. Since the DeepFakeDetection data set contains videos, we first need to extract the frame and then crop the inner faces from them. Due to the different content of the scenes of the videos, these cropped faces have different sizes.

The pre-processing part from the pipeline is size independent, thus no changes are required. However, this is not true for the classifiers, since they expect a fixed amount of features. Therefore, we have added an extra processing block just before the classifier that interpolates the 1D Power Spectrum to a fix size (300) and normalizes it dividing it by the 0th frequency component. The rest of the pipeline remains unchanged.

3) Method 1D Power Spectrum: As in the previous experiments, Fig. 7 shows that deepfake images have a noticeably

	80% (train) - 20% (test)	
# samples	SVM	Logistic Reg.
2000	87%	81%
1000	87%	78%
200	86%	70%
20	82%	35%

TABLE VII: Test accuracy using SVM classifier and logistic regression classifier under different data settings.

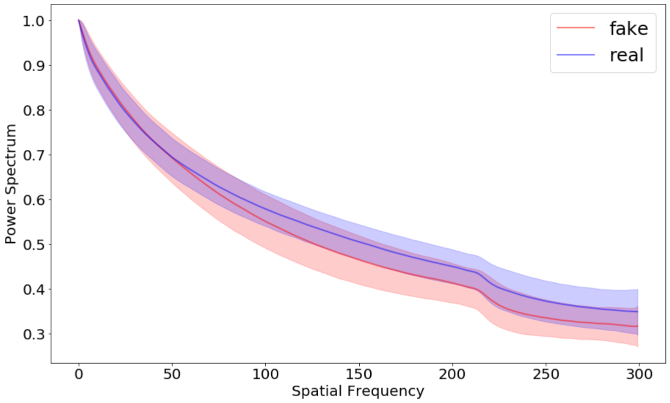


Fig. 8: 1D Power Spectrum statistics from DeepFakeDetection data set.

different frequency characteristic. Despite of having a similar behaviour along the spatial frequency, there is a clear offset between the real the fakes that allows the images to be classified.

Table VII contains the classification accuracy for the supervised algorithms. These results confirm the robustness of frequency components as classification features. Nevertheless, in this case, we have observed a slightly different behaviour with respect to *Faces-HQ* accuracy results (see Table II). The problem is become harder for low-resolution inputs. Hence, the accuracy starts to decrease when the number of samples is smaller than 1000, specially, for the logistic regression.

The dependency on samples and the non-perfect classification accuracy can be understood by looking at Fig. 8. We can see how the standard deviations from the real and the deepfake statistics overlap with each other, meaning that some samples will be misclassified. As a result, it is not recommendable to reduce the number of features, since now the classifiers are much more sensitive to the number of features.

Finally, we compute the average classification rate per video, applying a simple majority vote over the single frame classifications. Table VIII shows the accuracy test results, which are relatively higher than the previous ones based on a frame by frame evaluation.

SVM	Logistic Reg.
91%	82%

TABLE VIII: Test accuracy per video using SVM classifier and logistic regression classifier.

V. Discussion and Conclusion

In this paper, we described and evaluated the efficacy of a new method to expose AI-generated fake faces images. Our approach is based on a high-frequency component analysis. We performed extensive experiments to demonstrate the robustness of our pipeline independently of the source image. We show that our method is able to detect high- and medium-resolution deepfake images on two data sets with data from various GANs with 100% accuracy. Low-resolution content is harder to identify since the available frequency spectrum is much smaller. Nevertheless, we are able to identify low-resolution fakes in a popular benchmark with 91% accuracy.

References

[1] 100,000 faces generated. https://generated.photos/.

[2] D. Afchar, V. Nozick, J. Yamagishi, and I. Echizen. Mesonet: a compact facial video forgery detection network. In *2018 IEEE International Workshop on Information Forensics and Security (WIFS)*, pages 1–7. IEEE, 2018.

[3] B. E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152. ACM, 1992.

[4] A. Brock, J. Donahue, and K. Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

[5] M. Brundage, S. Avin, J. Clark, H. Toner, P. Eckersley, B. Garfinkel, A. Dafoe, P. Scharre, T. Zeitzoff, B. Filar, et al. The malicious use of artificial intelligence: Forecasting, prevention, and mitigation. *arXiv preprint arXiv:1802.07228*, 2018.

[6] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[7] D. Cozzolino, D. Gragnaniello, and L. Verdoliva. Image forgery localization through the fusion of camera-based, feature-based and pixel-based techniques. In *2014 IEEE International Conference on Image Processing (ICIP)*, pages 5302–5306. IEEE, 2014.

[8] D. Cozzolino and L. Verdoliva. Noiseprint: a cnn-based camera model fingerprint. *IEEE Transactions on Information Forensics and Security*, 2019.

[9] T. J. De Carvalho, C. Riess, E. Angelopoulou, H. Pedrini, and A. de Rezende Rocha. Exposing digital image forgeries by illumination color classification. *IEEE Transactions on Information Forensics and Security*, 8(7):1182–1194, 2013.

[10] P. Ferrara, T. Bianchi, A. De Rosa, and A. Piva. Image forgery localization via fine-grained analysis of cfa artifacts. *IEEE Transactions on Information Forensics and Security*, 7(5):1566–1577, 2012.

[11] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[12] D. Güera and E. J. Delp. Deepfake video detection using recurrent neural networks. In *2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6. IEEE, 2018.

[13] C.-C. Hsu, C.-Y. Lee, and Y.-X. Zhuang. Learning to detect fake face images in the wild. In *2018 International Symposium on Computer, Consumer and Control (IS3C)*, pages 388–391. IEEE, 2018.

[14] T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

[15] T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4401–4410, 2019.

[16] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[17] H. Li, B. Li, S. Tan, and J. Huang. Detection of deep network generated images using disparities in color components. *arXiv preprint arXiv:1808.07276*, 2018.

[18] Y. Li, M.-C. Chang, and S. Lyu. In ictu oculi: Exposing ai generated fake face videos by detecting eye blinking. *arXiv preprint arXiv:1806.02877*, 2018.

[19] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *Proceedings of the IEEE international conference on computer vision*, pages 3730–3738, 2015.

[20] L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[21] F. Marra, D. Gragnaniello, D. Cozzolino, and L. Verdoliva. Detection of gan-generated fake images over social networks. In *2018 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*, pages 384–389. IEEE, 2018.

[22] S. McCloskey and M. Albright. Detecting gan-generated imagery using color cues. *arXiv preprint arXiv:1812.08247*, 2018.

[23] X. Pan, X. Zhang, and S. Lyu. Exposing image splicing with inconsistent local noise variances. In *2012 IEEE International Conference on Computational Photography (ICCP)*, pages 1–10. IEEE, 2012.

[24] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[25] A. Rössler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner. FaceForensics++: Learning to detect manipulated facial images. In *International Conference on Computer Vision (ICCV)*, 2019.

[26] X. Yang, Y. Li, and S. Lyu. Exposing deep fakes using inconsistent head poses. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8261–8265. IEEE, 2019.

[27] N. Yu, L. Davis, and M. Fritz. Attributing fake images to gans: Learning and analyzing gan fingerprints. In *International Conference on Computer Vision (ICCV)*, October 2019.

[28] P. Zhou, X. Han, V. I. Morariu, and L. S. Davis. Two-stream neural networks for tampered face detection. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1831–1839. IEEE, 2017.