Forensic Analysis of GAN Training and Generation: Output Artifacts Assessment of Circles and Lines

Stefan Seidlitz and Jana Dittmann
Department of Computer Science
Otto-von-Guericke University
Magdeburg, Germany
e-mail: stefan.seidlitz@ovgu.de | jana.dittmann@iti.cs.uni-magdeburg.de

***Abstract*—Motivated by the challenges in different forensic detection tasks based on machine learning, this paper evaluates the training behavior, as well as the generation performance of images which are generated by Generative Adversarial Networks (GANs) based on simple geometric shapes using the example of circles and lines. Circles, for example, are relevant for DeepFaceFakes where eyes might be checked for inconsistencies. Therefore, we trained several StyleGAN3 models with different self-created training data sets using geometrical shapes of circles and lines. We use these models to generate fake circles and lines from a random latent vector, which we then forensically analyzed in two different ways: a visual, subjective evaluation based on an observation as well as an automated Circle-Checking approach. In both experiments, we were able to show on the example of StyleGAN3, that generative approaches have difficulties with the generation of geometric shapes: circles are often more comparable to eggs, lines are mostly not linear. Our contribution is to advance the knowledge on what kind of artifacts a Generative Adversarial Network generates. This gives a first tendency for new detection strategies to identify these artifacts, based on geometrical shapes in DeepFake images.**

Keywords-Media Forensic; Generative Adversarial Networks (GAN); DeepFake; Advance and Challenges.

I. Introduction and Our Concept

Motivated by the increasing use of Artificial Intelligence (AI) in media creation and processing, as well as for security incident detection in malicious cases, forensic explainability and reproducibility of the process is necessary for understanding and evaluating the results. In our paper, we propose to use a forensic analysis by using simplified and well-defined shapes on the example of geometric shapes of circles and lines, available as Open Data in [2]. The goal is to train models with a set of simplified and well-defined shape images to measure and study the output from the generation. The well-defined training guides the comparison and allows measuring artifacts in the output, which were not included in the training. In the comparison, we use a visual human-based assessment and a first, straight forward automated analysis on the example of 4 circle data sets. The goal is to show a possible first setting of the idea of simplified and well-defined shapes with a first tendency of results advancing the knowledge.

In particular, we are motivated by the DeepFakes cases. For example, recent methods are AutoEncoders (AEs) based approaches to replace the face or voice from person with the face or voice of another person. Further, we designate full synthetic, non-existing faces which are generated by a Generative Adversarial Network (GAN) also as DeepFakes. To avoid a critical use of DeepFakes a detection of DeepFakes is necessary in specific cases. In addition, the identification of characteristic DeepFake traces improves the examination as an essential part of a forensic investigation and it is required to showcase evidence in court. Nevertheless, full synthetic GAN based DeepFake images consist of a face in the foreground as well as a generated background area. While the faces in a DeepFake usually look deceptively real, the background can often be a strong indication of a DeepFake. Here, often the model is unable to generate geometric patterns, as it is not considered in the original training data. In consequence, there are several background artifacts e.g. letters consisting of curved lines in the background.

Most approaches like [3] or [4] for the detection of DeepFakes fall back to the use of AI, especially Neural Networks (NNs). Guo et al. [5] shows that the shape of the pupils in GAN-generated faces is irregular without giving any explanations of the causes. It is unclear to us whether other shapes in a facial image have an influence on the training behavior of a GAN. But, the observation of Guo et al. motivated us to train a StyleGAN3 [6] only with images of simplified shapes using the GitHub implementation of [1]. The National Institute of Standards and Technology (NIST) [7] suggests explanations in *purposes* and *styles*. Our approach has to address this and we follow the two main objectives: (A) How accurate can a GAN be used to generate perfect shapes? and (B) In a mixed training scenario, has a shape type an influence to another (different) shape type?

First, we summarize the key aspects of the development process of ProgressiveGAN [8], StyleGAN [9] and its extensions and introduce our approach in the next subsections. After a small overview about the implementation in Section II, we describe in Section III our observations on our training iterations separated on general observations as well as specific training observations. Additionally, we compare and discus the Automated Circle-Checking with our visible observations on all circle images. Finally, we conclude our paper in Section V.

A. Used methods from State of the Art

Goodfellow et al. [10] create with GANs the fundamentals for the work of Karras et al. [8]. This work allows

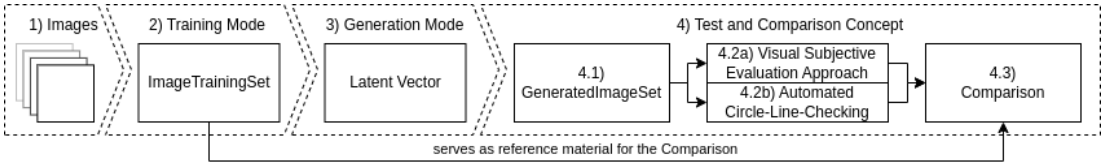


Figure 1. Pipeline for our approach, with StyleGAN3 implementation from [1].

the generation of high-resolution digital images, especially artificial face images. With ProgressiveGAN, they expand the trained image size step by step over the whole training phase. Additionally, they created a new image based high-quality data set named CelebA-HQ. In Karras et al. [9], the authors replaced the traditional generator network with a mapping network followed by a synthesis network, which allows the style transfer between latent vectors. However, the used Adaptive Instance Normalization (AdaIN) within the synthesis networks caused artifacts looking like water droplets in the generated images. Instead of the Instance Normalization, Karras et al. used a demodulation technique which results in normalized weights within the synthesis network [11]. Further, the technique of the progressive growing resolution in [8] results in a strong location preference for details. Instead of the progressive growing approach, the authors of StyleGAN2 used a skip generator and a residual discriminator architecture. Karras et al. [12] stabilized the discriminator of StyleGAN2-ada to avoid overfitting with different augmentation techniques which allows training the GAN with less training data. With StyleGAN3, Karras et al. [6] redesigned the generator network to avoid aliasing artifacts, specific details on its configuration are given in [6].

Those and further DeepFake generation methods like few-shot vid2vid [13] or Collaborative Diffusion [14] are used for the DF40 data set [15]. The aim of this data set is to combine the different DeepFake techniques such as face-swapping, face-reenactment and entire image synthesis. In consequence, the challenging task to tackle the generalization problem in DeepFake detection is opened, because most DeepFake detection strategies are focused to detect specific DeepFake techniques.

B. Our Approach

We divide our approach into three phases which are described in the following sections. Additionally, we highlight the main aspects of our approach in Figure 1.

1) Training Mode: For the *Training Mode*, different data sets called *ImageTrainingSet* are created which are described in Table I and some example images are visualized in Table III. Each data set consists of 50k images which have a shape of 64×64 pixels. Only grayscale images are created. The background color of each image is set to white (color value: 255). All geometric objects use a black color (color value: 0). We mainly decide between 'single' and 'multi' *ImageTrainingSet* which address the amount of geometric objects within the images. Images in a 'single' data set have only one geometric object (Table I: ID 1, 3, 4 and 6) whereas images in a 'multi' data set have between one and ten geometric objects (Table I: ID 2 and 5). Currently, no image has two different kinds of geometric objects (circles and rings or lines). The position of every geometric object is set randomly. But no object was allowed to have connections with the image boundary or with other objects in the same image. In all images of every *ImageTrainingSet*, we define a distance of 2 pixels to the boundary of the image. In the *multi ImageTrainingSet* of circles, we define a minimum distance of 2 pixels to other circles. The size of the circled shapes is between 3 and 60 pixels. Further, we create circled rings with a border size of 1 to 5 pixels, the ring size is the same as full circles. In case of lines, we set the minimal size to 10 pixels, the maximum size is calculated in combination with the boundary pixels and their orientation within the image. Each line has a thickness of

TABLE I. OVERVIEW OF ALL CREATED TESTING DATA SETS FOR STEP 1 (COMPARE TO FIGURE 1).

ID	Type	Content	Image Count	Image Size	Image Color	Number and Type of Shapes per Image
1	single circle	black circles and circled black rings	50.000	64×64	black / white	only one with random size and position, no connection with border
2	multiple circle	black circles and circled black rings	50.000	64×64	black / white	between one and ten with random size and position, no connection between other circles and with border
3	single horizontal line	black horizontal line	50.000	64×64	black / white	only one with random size and position, no connection with border
4	single line	black line	50.000	64×64	black / white	only one with random size, line direction and position, no connection with border
5	multiple lines	black lines	50.000	64×64	black / white	between one and ten with random size, line direction and position, no connection with other lines and with border
6	single circle or line	black circles rings, and lines	circles and rings: 25.000; lines: 25.000	64×64	black / white	only one with random size, direction and position, no connection with border; sub sets are randomized reused from data set 1 and 4

1 pixel. All other parameters for the line data sets are similar to the circle data sets. Please note, in a *multi ImageTrainingSet* the shape size is also defined from other shapes, because in an image no connections to other shapes are allowed. For the *combined ImageTrainingSet* of circles and lines, we reuse subsets of both *multi ImageTrainingSet* for circles and lines. The images of both subsets are selected randomly.

2) Generation Mode: In the *Generation Mode* we differentiate between two generation methods. The first generation method is done during the training process of StyleGAN3. It creates after a defined time of training steps a snapshot of the current training state as well as an overview image of different generated images. This overview image is a grid of 32×32 images, where each image is generated from a random selected latent vector. Separately, from selected snapshots of several models we generate images from a latent vector defined by a seed using the image generation script gen_images.py of [1]. The seed is chosen by randomly generated numbers between 0 and 10000 using a bash script which executes the generation script of StyleGAN3. The script will be available at [2].

3) Test and Comparison Concept: We propose in our concept a **Visual Subjective Evaluation Approach** on circles and lines to support explainability for humans. Our criteria are (derived from Visual Morphing Detection Assessment [16]): (1) How homogeneous are the colors of the background or the generated geometrical objects? (2) Are there visible differences between the generated and the ideal geometrical object (e.g., missing symmetry)? (3) Is the amount of objects within the generated images similar to the specific training set? (4) Are there intersections with other generated objects or the border of the image?

Furthermore, we suggest to use for circles an automated approach (**Automated Circle-Checking**) by using Hough Transform. Here, we show in a first test setting for circles an approach for their detection. The subjective and automated results are compared for a circle test case to show possibilities and limitations by deriving further work.

TABLE II. STYLEGAN3 TRAINING CONFIGURATION (STEP 2, FIGURE 1).

Required		
outdir	*path to the output directory*	
cfg	*stylegan3-t*	
data	*path to the training data set*	
GPUs	1	(proposed values from the Readme file for the training parameter)
batch	32	
gamma	8.2	
Optional features		
mirror	1	
all other parameters are not set, either the default configurations were used or the specific parameter was not used here		
Misc hyperparameters		
those parameters are not set, either the default configurations were used or the specific parameter was not used here		
Misc settings		
kimg	25000 (default value)	
snap	50 (default value) or 10	
all other parameters are not set, either the default configurations were used or the specific parameter was not used here		

II. IMPLEMENTATION

A. Preparation for the data sets

For the creation of our data sets, we use Python and the programming library Pillow, which is needed for the creation of circles and rings (please note: for the creation of circles, the function for the creation of ellipse is needed) or lines. Other geometric shapes are not used for this work. Table I shows the specific content of all available data sets. Note, the data set ID 6 reused randomly 50 % of the images of the data set with the ID 1 and 50 % of the images of the data set with the ID 4. Some example images for each data set are given in Table III. The data sets are available at [2].

All images are created as BMP pixel image without compression. For the training, the data sets need to be preprocessed with the dataset_tool of [1] which converts all images into PNG images. Furthermore, this tool creates a ZIP archive that includes all images of the specific data set.

B. Training approaches and expected outcome

Every data set was used to train two StyleGAN3 models whereby only the snap parameter was changed to store the models more frequently. The reason for this is on the one hand to ensure reproducibility and on the other hand to get a finer view how the models are changing over the training time. Because of the color space of the images, StyleGAN3 automatically chooses the shape $[1, 64, 64]$ where the first value addresses the color space and the last two values the dimensions of the images. In consequence StyleGAN3 identified automatically the correct color space of the training image data set. All other parameters are set on the basis of the "Training" section of the README.md file of the GitHub repository of StyleGAN3 [1]. Compare here also the Table II. Both models from the same data set were trained parallel on one separate GPU card of the same type. The final trained models are available at [2].

Table IV describes the expected results of each training approach to imitate the given image data sets. Note, the snapshot frequency (third column of Table IV) is calculated from the snap parameter with the tick parameter which is default set to the value 4.

For this paper, the evaluation was performed manually by a human with a visual observation of randomly generated images by the StyleGAN3 generator. For this purpose, the automatically generated grid images as well as self generated images are used.

III. EVALUATION

A. Subjective Evaluation Approach: Observations on StyleGAN3 generated shapes

We randomly analyzed samples from every training approach over different training iterations. Thereby, several unexpected not trained lines and circles (defined as errors) in all generated images are observed. We distinguish those errors between general and specific errors which are described in following sections: The general observations describe errors

TABLE III. EXAMPLE IMAGES FOR ALL DATA SETS FROM TABLE I, SEE STEP 1 OF FIGURE 1.

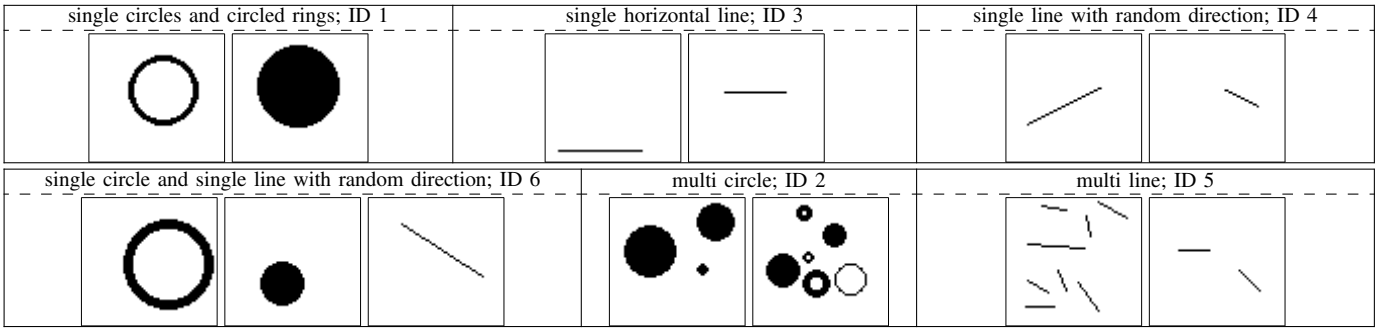


TABLE IV. TRAINING INTENTIONS, EVERY MODEL IS TRAINED WITH THE CONFIGURATION OF TABLE II, ONLY THE SNAPSHOT SEQUENCE WAS CONFIGURED WHICH IS GIVEN IN THE COLUMN "SNAPSHOTS" (SEE STEP 2 OF FIGURE 1).

ID	used data set	snapshots	training intention / expected training behavior
1	multi: circle & rings; data set id: 2	every 200 kimg	Size, shape and color of circles and ring should be similar to the training data set. The generator should create between 1 and 10 objects (circles and/or rings) to emulate images from the data set. No object should have connections with the border of the image or with other objects.
2		every 40 kimg	
3	single: circle & rings; data set id: 1	every 200 kimg	Size, shape and color of circles and ring should be similar to the training data set. The generator should create only 1 object (circle or ring) to emulate images from the data set. No object should have connections with the border of the image.
4		every 40 kimg	
5	single: horizontal lines; data set id: 3	every 200 kimg	Size, shape, alignment and color of lines should be similar to the training data set. The generator should create only 1 horizontal line to emulate images from the data set. No line should have connections with the border of the image.
6		every 40 kimg	
7	single: lines with a random direction; data set id: 4	every 40 kimg	Size, shape, alignment and color of lines should be similar to the training data set. The generator should create only 1 line with an indifferent alignment to emulate images from the data set. No line should have connections with the border of the image.
8		every 200 kimg	
9	multi: lines with a random direction; data set id: 5	every 200 kimg	Size, shape, alignment and color of lines should be similar to the training data set. The generator should create between 1 and 10 lines with an indifferent alignment to emulate images from the data set. No line should have connections with the border of the image or intersections/connections with other lines.
10		every 40 kimg	
11	single: circles, rings & lines; data set id: 6	every 40 kimg	The behavior of this training test should be similar to the training IDs 3 and 4 in combination to 7 and 8. The influence from specific features of one training set to the other training set is unexpected before the training process starts.
12		every 200 kimg	

which are present in all images over all training approaches and specific observations which are only present in specific training approaches.

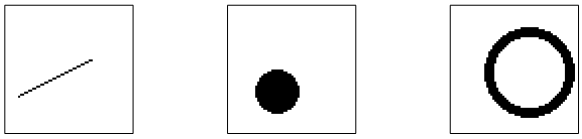


Figure 2. Scaled color scheme on real images using the image processing tool Gimp (which have no effect here).

1) General observations: We noticed in every generated image that areas of the same visible color (e.g., background as well as geometric objects) are not homogeneous. For example, we identified on the white background color areas of bright gray colored pixels which are not visible at the first glance. For visualization reasons and because of comprehensibility aspects, we scale the color values on Figure 2 and Figure 3 using Gimp [17]. Figure 3 visualized these 'invisible' artifacts. Because of the BMP input file type, these artifacts are not explainable with our training data sets (see images in Figure 2).

Figure 3. Scaled color scheme on fake images using the image processing tool Gimp.

In case of all training approaches which used a 'single' data set for the training (Table IV: ID 3 to 8, 11 and 12), some images are generated where more than one geometric object is created. In Table VII there are some images of a 'single' approach shown where not exactly one geometric object was generated; especially for ID 1 image 1 and 3 with two geometric objects, for ID 3 image 2 with no geometric object, for ID 4 image 3 and 4 with two or three geometric objects and for ID 6 image 2 with two geometric objects. The reason for this can be caused by the latent vector or style vector of StyleGAN3. Because of that the influence of the random seeds to the latent vector as well as the latent vector to the image generation process should be analyzed. Further, it is possible that also generated images of all multi

training approaches are affected with a similar issue (Table IV: ID 1, 2, 9 and 10). The network is allowed to generate up to ten geometric objects, but it also not able to count the geometric objects which can be result in the generation of more than ten geometric objects. This circumstance is less visually recognizable than on a training approach with a single based data set, why a visual observation is here not effective.

2) Specific observations: The specific observation is divided into shape type of the geometric objects: circles and circled rings, lines and the mixed data set of lines and circles or circled rings (Table IV: ID 11 and 12). A separated observation of single and multi data sets is not useful because we noticed the same errors on both approaches. Compare here also Table V, which summarizes our observations for each training approach, and Table VII, which shows obvious errors.

The training approaches of both circle data sets (Table IV: ID 1 to 4), which will only address circles and circled rings, shows that StyleGAN3 was not able to train with all training iterations perfect circles or circled rings. Most generated circles do not show (because of the gridded circles, even if approximated) a specific rotational symmetry. Also, the sizes of the generated rings are not always the same on every position of the ring.

With respect to the line based data sets (Table IV: ID 5 to 10), only the generated horizontal lines (Table IV: ID 5 and 6) are most similar to their training data set. However, they have the same errors which are introduced in the previous Subsection III-A1: mostly on the line ends the color value is not always 0 (black). Once the lines were randomly rotated in their training data set, StyleGAN3 was not able to create perfectly straight lines. The generated lines (Table IV: ID 7 to 10) are curved, sometimes only one time, sometimes two times.

We also tried to train 25k single random line images with 25k single circle images at the same time (Table IV: ID 11 and 12). On this approach, we identified the same errors which are described before for the other training approaches. Only a transfer of features from circles or rings to lines or vice versa was not determined. Especially small lines and small circles seem to be mixed during the training process of StyleGAN3.

B. Objective Evaluation with the Automated Circle-Checking and its comparison with the Subjective Approach

For the Automated Circle-Checking, we use the Hough Circle Detection on our real- as well as fake Multi Circle data set (Table V, ID 1-4). As this detector is still in its early stages, it is currently unable to distinguish between genuine and fake images. The decision whether it is a real or a fake image is made by the examiner. We evaluate the detected position and size of our circles.

In most cases, the position of our real circles is detectable. Only the size was not always detected correctly. The reason for this could be due to the parameterization of the Hough transformation. Nevertheless, we have also tested our Automated Circle-Checking on our Multi Circle data set with fake circles. Please note that our fake images have a different color distribution: it is possible that the StyleGAN3 generator used any gray value between 0 and 255, while on real images only color values of 0 (circle color) and 255 (background) are possible. This fact has an influence on the circle detection using Hough transform. The circle detector was able to detect many small circles, which are usually a part of a large circle. Detection of the entire circle was rare. In some cases, the detector detects circles on a position where no circle was present. Table VI illustrates the behavior of the Automated Circle-Checking on real images compared to fake images.

IV. Discussion

Using the example of circles and lines, we could see that GANs are not able to generate exact circles or lines from a randomly given latent vector. This statement is further verified by our subjective in case of circles and lines as well as automated evaluation in case of circles. In relation to a more complex scenario, also in real faces specific shapes are given. Best example is the iris and the pupil of the eyes which is shown in the image process pipeline for an iris detection on Figure 4. Are there similar shape artifacts in full synthetic face images given and can our findings be used to identify other artifacts in synthetic face images? A first assessment is given in Figure 5. It is obvious that the extracted iris of a fake face generated by StyleGAN2 (extracted from

TABLE V. Visual observed errors of all trainings (see 4.2a of Fig. 1), objective evaluation performed with ID 1-4 and confirm errors.

ID	used data set	human observations
1 2	circle and rings (multi); data set id: 2	no homogeneous geometric area, no given symmetry, no circle shape, objects also in area of border possible
3 4	circle and rings (single); data set id: 1	no homogeneous geometric area, no given symmetry, no circle shape, objects also in area of border possible, sometimes more than one object
5 6	lines (horizontal); data set id: 3	lines have mostly the same horizontal direction, pixel values of lines are mostly homogeneous, only at the line border are different pixel values possible
7 8	lines (single, random direction); data set id: 4	non straight lines, partially one to two turning points on the line, smoother transitions due to gray value change on line segments
9 10	lines (multi, random direction); data set id: 5	same visible observation like ID 7 or ID 8; line alignments are similar to the line alignments of the initial data set (compare table I ID 5)
11 12	circles, rings and lines (single); data set id: 6	shapes have the same errors which are described for the training runs of ID 3, ID 4, ID 7 and ID 8; a feature transfer (or error transfer) from line to circle and vice versa are not visible

thispersondoesnotexist.com in May 2024) does not have a typical circular shape which was confirmed by our automated detection approach.

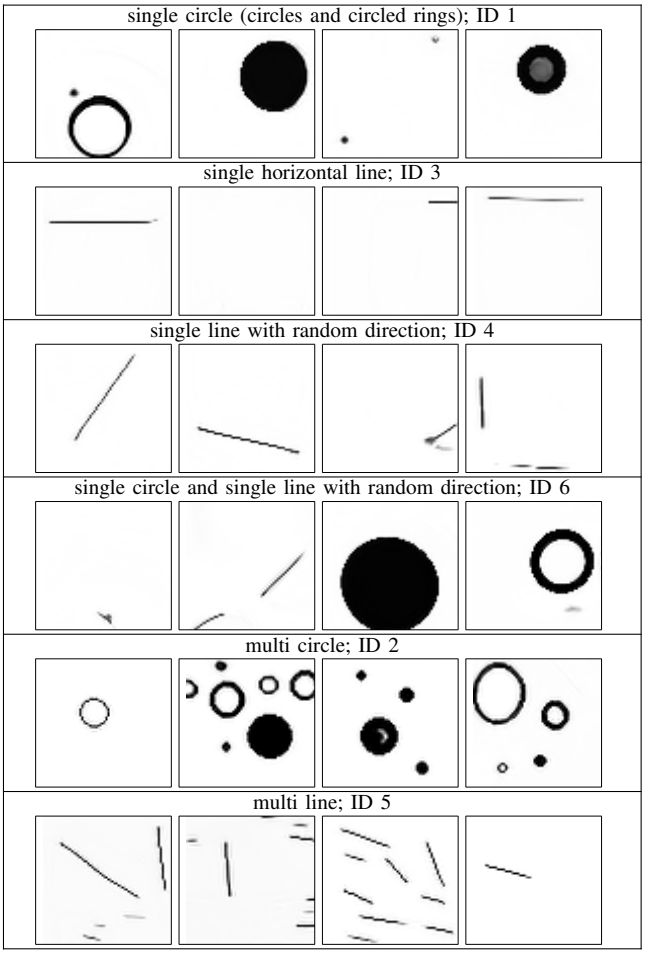
Figure 4. Circle detection on an eye of a real person using the London Face Set [18].

Figure 5. Circle detection on an eye of a fake person generated by StyleGAN2 using the web page https://thispersondoesnotexist.com/.

At the moment, we evaluate our approach mostly on small images of real or generated geometrical shapes like circles or lines. Against, the experiments on facial images were performed only on a few images which results in new challenges compared to our initial geometrical data sets. In this case, it is necessary to have more pre-processing steps until the geometrical shape evaluation is usable. Furthermore, the image size of synthesized face images is usually larger than the images in our data sets, which requires different parameter settings for the Hough transform than those used for small images.

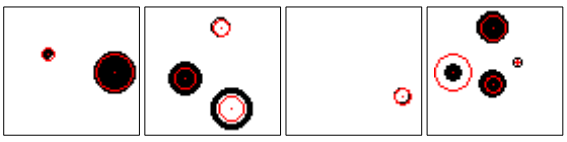
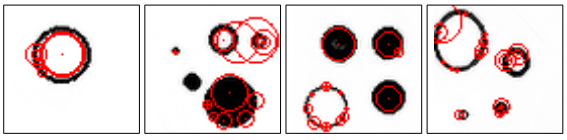
Additionally, our approach is only tested on images with a small size of 64×64 pixels which limits specific attributes of the geometric shape. Because of the rasterized pixel graphic, circles are limited in their minimum size. If they are too small, they are not distinguishable from squares or rhombuses. On larger images, large rasterized circles are more comparable to real circles than on smaller images. This allows a calculable metric which can be used for an identification of DeepFake images in future work.

TABLE VII. GENERATED EXAMPLE IMAGES (ONLY GRAYSCALE) FOR ALL DATA SETS WHICH ARE SHOWN IN TABLE I (SEE STEP 4.1 OF FIGURE 1).



In difference to circles, straight lines within the facial area are very unusual. But the detection is not limited to the face. As introduced, especially for full synthetic DeepFake images also the background can be used for their detection. The probability for possible fake lines is in the background area of DeepFake images higher. Further, the use case of the geometrical shape detection does not need to be restricted to DeepFakes. Han et al. [19] propose a GAN for

TABLE VI. COMPARISON OF AUTOMATED CIRCLE DETECTION (FIG. 1 4.2B) FOR REAL AND FAKE IMAGES, RED LINES HIGHLIGHTS THE DETECTED CIRCLES, HUMAN BASED COMPARISON BETWEEN THE VISUAL AND AUTOMATED APPROACH CONFIRM ERRORS IN GAN IMAGE GENERATION (FIG. 1 4.3).

automated detection on **real** circles and circled rings);		Most circles from training are automated detected correctly in their position and size. In General the detected center position is identical with the given center position. The automated detection of the circle size is not always the same compared to the visual circle.
automated detection on **fake** circles and circled rings);		Automated detection of small circles from fake circle generation was mostly correct in their position and size. On bigger circles more than one circle were detected. Specially the detection of big circled rings results in the detection of small circles on the border area of those rings. The detection of the whole ring was not successful. There are also shapes which was not detected as circle, but visually also not identified as circle by the human.

the generation of synthetic license plates of cars. Within the European Union, all license plates follow a specific standard. Our circle and line detection approach offers further applications. For example, the Germany License Plate DIN standard DIN 74069:2022-10 [20] with specific regularities for plate size and position as well as color, font and character size offers an evaluation: Would a GAN be able to correctly reproduce a German license plate and would be a GAN able to be detected a synthetic generated plate as DeepFake?

V. Conclusion

In this work, we introduced a new methodology to bring GANs to their limit by reducing the level of detail of the training material. We follow the challenge to identify issues which are a result of its generation process. To be specific, we train different StyleGAN3 models with different gray scale data sets of images with geometric shapes. On this way, our approach shows the root cause of circle artifacts in GAN generated data and explains therefor findings the artifact results in Guo et al. [5]. For further investigation on this question, we currently enhance our Automated Circle-Checking to an Automated Circle-Line-Checking approach. Additionally, we want to establish our approach to other generative AI technologies such as AEs.

As our approach uses StyleGAN3 implementations from [6] it can also be used in further decision support systems for DeepFake detection purposes. Further, both our generator and detector are provided as open source models and open data [2] to allow transparency in understanding and reproducing the inner processes of models. For future explainability, we plan to introduce on the one hand an improved user based comparison of potential fake and ideal circles and on the other hand a quantitative evaluation method with the definition of scores based on the overlapping areas of original and reproduces circles.

Acknowledgments and Author Contributions

The work in DeepFakes in this paper is funded in part by the German Federal Ministry of Education and Research (BMBF) under grant number FKZ: 13N15736 (project "Fake-ID"). The research for transparency with Open Source and Open Data conducted within this paper was partly funded by the European Union Project "CyberSecurity-Verbund LSA II" (Grant No.: ZS/2023/182058) - "CyberSecurity-Verbund LSA II – Prävention, Detektion und Reaktion mit Open Source-Perspektiven".

Initial idea & methodology: J. Dittmann (JD); Conceptualization: S. Seidlitz (StS) and JD; Modeling & application in the context of DeepFake: StS; Writing – original draft: StS and JD; Writing – review & editing: StS and JD. Thanks to Nischay Purnekar (University of Siena, Italy) for proof reading.

References

[1] T. Karras *et al.*, *GitHub repository of StyleGAN3*, https://github.com/NVlabs/stylegan3, Oct. 29, 2024.

[2] S. Seidlitz and J. Dittmann, *Specific Scripts, Models and Data of this Paper*, https://cloud.ovgu.de/s/sBRWzxSLYikp64x, Oct. 2, 2024.

[3] L. Nataraj *et al.*, "Detecting GAN generated fake images using co-occurrence matrices," *Electronic Imaging*, vol. 31, no. 5, pp. 532-1–532-1, 2019. DOI: 10.2352/ISSN.2470-1173.2019.5.MWSF-532.

[4] F. Marra, C. Saltori, G. Boato, and L. Verdoliva, "Incremental learning for the detection and classification of GAN-generated images," *2019 IEEE International Workshop on Information Forensics and Security (WIFS)*, pp. 1–6, 2019.

[5] H. Guo, S. Hu, X. Wang, M.-C. Chang, and S. Lyu, "Eyes tell all: Irregular pupil shapes reveal GAN-generated faces," in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 2904–2908. DOI: 10.1109/ICASSP43922.2022.9746597.

[6] T. Karras *et al.*, "Alias-free generative adversarial networks," *CoRR*, vol. abs/2106.12423, 2021. arXiv: 2106.12423.

[7] P. J. Phillips *et al.*, "Four principles of explainable artificial intelligence," Tech. Rep., Sep. 2021. DOI: 10.6028/nist.ir.8312.

[8] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of GANs for improved quality, stability, and variation," *CoRR*, vol. abs/1710.10196, 2017. arXiv: 1710.10196.

[9] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," *CoRR*, vol. abs/1812.04948, 2018. arXiv: 1812.04948.

[10] I. J. Goodfellow *et al.*, *Generative adversarial networks*, 2014. arXiv: 1406.2661 [stat.ML].

[11] T. Karras *et al.*, "Analyzing and improving the image quality of stylegan," *CoRR*, vol. abs/1912.04958, 2019. arXiv: 1912.04958.

[12] T. Karras *et al.*, "Training generative adversarial networks with limited data," *CoRR*, vol. abs/2006.06676, 2020. arXiv: 2006.06676.

[13] T.-C. Wang *et al.*, "Few-shot video-to-video synthesis," in *Conference on Neural Information Processing Systems (NeurIPS)*, 2019.

[14] Z. Huang, K. C. Chan, Y. Jiang, and Z. Liu, "Collaborative diffusion for multi-modal face generation and editing," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

[15] Z. Yan *et al.*, "Df40: Toward next-generation deepfake detection," *arXiv preprint arXiv:2406.13495*, 2024.

[16] A. Makrushin, D. Siegel, and J. Dittmann, "Simulation of border control in an ongoing web-based experiment for estimating morphing detection performance of humans," in *Proceedings of the 2020 ACM Workshop on Information Hiding and Multimedia Security*, ser. IH&MMSec '20, Denver, CO, USA: Association for Computing Machinery, 2020, pp. 91–96, ISBN: 9781450370509. DOI: 10.1145/3369412.3395073.

[17] The GIMP Development Team, *Gimp*, version 2.10.12, https://www.gimp.org, Jun. 12, 2019.

[18] L. DeBruine and B. Jones, "Face Research Lab London Set," May 2017. DOI: 10.6084/m9.figshare.5047666.v3.

[19] B.-G. Han, J. T. Lee, K.-T. Lim, and D.-H. Choi, "License plate image generation using generative adversarial networks for end-to-end license plate character recognition from a small set of real images," *Applied Sciences*, vol. 10, no. 8, 2020, ISSN: 2076-3417. DOI: 10.3390/app10082780.

[20] DIN 74069:2022-10, "Retroreflektierende Kennzeichenschilder, Stempelplaketten und Plakettenträger für Kraftfahrzeuge und deren Anhängefahrzeuge [in English: Retro-reflective registration plates, seal-sticker and sticker-cover for motor vehicles and their trailers]," DIN Media GmbH, Berlin, Tech. Rep., 2022. DOI: 10.31030/3379383.