https://doi.org/10.2352/ISSN.2470-1173.2021.4.MWSF-276

Detection, Attribution and Localization of GAN Generated Images

Michael Goebel; University of California, Santa Barbara; Santa Barbara, CA
Lakshmanan Nataraj, Tejaswi Nanjundaswamy, Tajuddin Manhar Mohammed, Shivkumar Chandrasekaran, B. S. Manjunath; Mayachitra Inc; Santa Barbara, CA

Abstract

Recent advances in Generative Adversarial Networks (GANs) have led to the creation of realistic-looking digital images that pose a major challenge to their detection by humans or computers. GANs are used in a wide range of tasks, from modifying small attributes of an image (StarGAN [14]), transferring attributes between image pairs (CycleGAN [92]), as well as generating entirely new images (ProGAN [37], StyleGAN [38], SPADE/GauGAN [65]). In this paper, we propose a novel approach to detect, attribute and localize GAN generated images that combines image features with deep learning methods. For every image, co-occurrence matrices are computed on neighborhood pixels of RGB channels in different directions (horizontal, vertical and diagonal). A deep learning network is then trained on these features to detect, attribute and localize these GAN generated/manipulated images. A large scale evaluation of our approach on 5 GAN datasets comprising over 2.76 million images (ProGAN, StarGAN, CycleGAN, StyleGAN and SPADE/GauGAN) shows promising results in detecting GAN generated images.

Introduction

The advent of Convolutional Neural Networks (CNNs) [43, 72] has shown application in a wide variety of image processing tasks, and image manipulation is no exception. In particular, Generative Adversarial Networks (GANs) [24] have been one of the most promising advancements in image enhancement and manipulation - the generative Artificial Intelligence (AI) patents grew by 500% in 2019 [2]. Due to the success of using GANs for image editing, it is now possible to use a combination of GANs and off-the-shelf image-editing tools to modify digital images to such an extent that it has become difficult to distinguish doctored images from normal ones. In December 2019, Facebook announced that it removed hundreds of accounts whose profile pictures were generated using AI [1, 3].

The GAN training procedure involves a generator and discriminator. The generator may take in an input image and a desired attribute to change, then output an image containing that attribute. The discriminator will then try to differentiate between images produced by the generator and the authentic training examples. The generator and discriminator are trained in an alternate fashion, each attempting to optimize its performance against the other. Ideally, the generator will converge to a point where the output images are so similar to the ground truth that a human will not be able to distinguish the two. In this way, GANs have been used to produce “fake” images that are very close to the real input images. These include image-to-image attribute transfer (CycleGAN [92]), generation of facial attributes and expressions (StarGAN [14]), as well as generation of whole new images such as faces (ProGAN [37], StyleGAN [38]), indoors (StyleGAN) and landscapes (SPADE/GauGAN [65]). In digital image forensics, the objective is to both detect these fake GAN generated images, localize areas in an image which have been generated by GANs, as well as identify which type of GAN was used in generating the fake image.

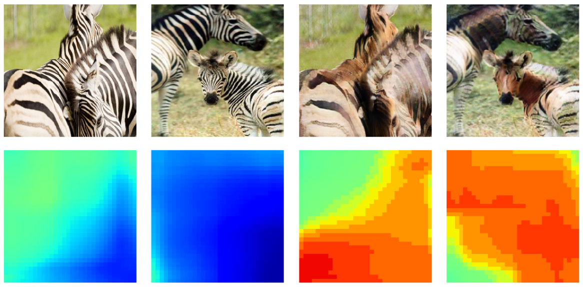


Figure 1: Input test set images on the top row, and our proposed detection heatmaps on the bottom. The two images on the left are authentic zebra images, those on the right are generated using CycleGAN.

In the GAN training setup, the discriminator functions directly as a classifier of GAN and non-GAN images. So the question could be raised as to *why not use the GAN discriminator to detect if it’s real or fake?* To investigate this, we performed a quick test using the CycleGAN algorithm under the maps-to-satellite-images category, where fake maps are generated from real satellite images, and vice versa. In our test, we observed that the discriminator accuracy over the last 50 epochs was only 80.4%. However, state-of-the-art deep learning detectors for CycleGAN often achieve over 99% when tested on the same type of data which they are trained [56, 62, 89]. Though the discriminator fills its role of producing a good generator, it does not compare performance wise to other methods which have been suggested for detection.

While the visual results generated by GANs are promising, the GAN based techniques alter the statistics of pixels in the images that they generate. Hence, methods that look for deviations from natural image statistics could be effective in detecting GAN generated fake images. These methods have been well studied in the field of steganalysis which aims to detect the presence of hidden data in digital images. One such method is based on analyzing co-occurrences of pixels by computing a co-occurrence matrix. Traditionally, this method uses hand crafted features computed on the co-occurrence matrix and a machine learning classifier such as support vector machines determines if a message is hidden in the image [74, 73]. Other techniques involve calculating image residuals or passing the image through different filters before computing the co-occurrence matrix [67, 23, 17]. Inspired by steganalysis and natural image statistics, we propose a novel method to identify GAN generated images using a combination of pixel co-occurrence matrices and deep learning. Here we pass the co-occurrence matrices directly through a deep learning framework and allow the network to learn important features of the co-occurrence matrices. This also makes it difficult to perform adversarial perturbations on the co-occurrence matrices since the underlying statistics will be altered. We also avoid computation of residuals or passing an image through various filters which results in loss of information. We rather compute the co-occurrence matrices on the image pixels itself. For detection, we consider a two class framework - real and GAN, where a network is trained on co-occurrence matrices computed on the whole image to detect if an image is real or GAN generated. For attribution, the same network is trained in a multi-class setting depending on which GAN the image was generated from. For localization, a network is trained on co-occurrence matrices computed on image patches and a heatmap was is generated to indicate which patches are GAN generated. Detailed experimental results on large scale GAN datasets comprising over 2.76 million images originating from multiple diverse and challenging datasets generated using GAN based methods show that our approach is promising and will be an effective method for tackling future challenges of GANs.

The main contributions of the paper are as follows:

- We propose a new method for detection, attribution and localization of GAN images using a combination of deep learning and co-occurrence matrices.
- We compute co-occurrence matrices on different directions of an image and then train them using deep learning. For detection and attribution, the matrices are computed on the whole image and for localization, the matrices are computed on image patches to obtain a heatmap.
- We perform our tests on over 2.7 million images, which to our knowledge, is the largest evaluation on detection of GAN images.
- We provide explainability of our approach using t-SNE visualizations on different GAN datasets.
- We show the method holds under both varying JPEG compression factors and image patch sizes, accommodating a range of real-world use cases.

Related Work

Since the seminal work on GANs [24], there have been several hundreds of papers on using GANs to generate images. These works focus on generating images of high perceptual quality [60, 68, 71, 34, 5, 26, 37], image-to-image translations [34, 86, 92], domain transfer [77, 41], super-resolution [44], image synthesis and completion [47, 33, 84], and generation of facial attributes and expressions [49, 66, 41, 14]. Several methods have been proposed in the area of image forensics over the past years [21, 54, 9, 48, 81]. Recent approaches have focused on applying deep learning based methods to detect tampered images [7, 8, 70, 11, 6, 17, 91].

In digital image forensics, detection of GAN generated images has been an active topic in recent times and several papers have been published in the last few years [56, 80, 46, 59, 45, 36, 78, 57, 61, 20, 87, 31, 62, 89, 82, 58, 4, 93, 30, 64, 88, 83, 40, 35, 10, 22, 27, 13, 10, 25]. Other similar research include detection of computer generated (CG) images [19, 85, 53, 69]

In [56], Marra et al. compare various methods to identify CycleGAN images from normal ones. The top results they obtained are using a combination of residual features [16, 17] and deep learning [15]. In [46], Li et al. compute the residuals of high pass filtered images and then extract co-occurrence matrices on these residuals, which are then concatenated to form a feature vector that can distinguish real from fake GAN images. In [89], Zhang et al. identify an artifact caused by the up-sampling component included in the common GAN pipeline and show that such artifacts are manifested as replications of spectra in the frequency domain and thus propose a classifier model based on the spectrum input, rather than the pixel input.

We had previously proposed a 3 channel co-occurrence matrix based method [62], and many other papers have shown the efficacy of this method in their experimental evaluations [51, 63, 79, 64, 88, 32, 55]. However, in this paper we compute co-occurrence matrices on horizontal, vertical and diagonal directions, as well as compute them on image patches, thus facilitating detection, attribution and localization of GAN generated images.

Methodology

Co-Occurrence Matrix Computation

The co-occurrence matrices represent a two-dimensional histogram of pixel pair values in a region of interest. The vertical axis of the histogram represents the first value of the pair, and the horizontal axis, the second value. Equation 1 shows an example of this computation for a vertical pair.

$$C_{i,j} = \sum_{m,n} \begin{cases} 1, I[m,n] = i \, and \, I[m+1,n] = j \\ 0, otherwise \end{cases} \tag{1}$$

Under the assumption of 8-bit pixel depth, this will always produce a co-occurrence matrix of size 256x256. This is a key advantage of such a method, as it will allow for the same network to be trained and tested on a variety of images without resizing.

Which pairs of pixels to take was one parameter of interest in our tests. For any pixel not touching an edge, there are 8 possible neighbors. We consider only 4 of these for our tests; right, bottom right, bottom, and bottom left. The other 4 possible pairs will provide redundant information. For example, the left pairs are equivalent to swapping the order of the first and second pixel in the right pair. In the co-occurrence matrix, this corresponds to a simple transpose. There are many subsets of these 4 pairs which could be taken, but our tests consider only a few; horizontal, vertical, horizontal and vertical, or all.

Before passing these matrices through a CNN, some preprocessing is done. First, each co-occurrence matrix is divided by its maximum value. Given that the input images may be of varying sizes, this will force all inputs into a consistent scale. After normalization, all co-occurrence matrices for an image are stacked in the depth dimension. In the example of an RGB image with all 4 co-occurrence pairs, this will produce a new image-like feature tensor of size 256x256x12. Figure 2 gives a visualization of this process.

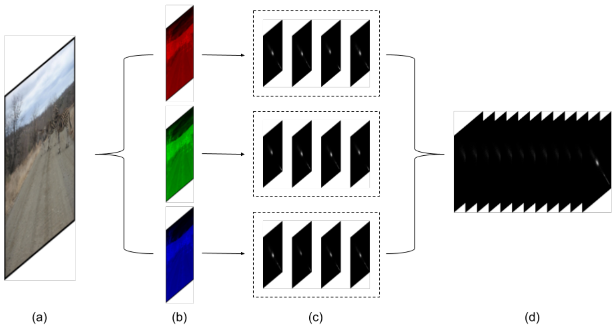


Figure 2: An example co-occurrence computation. The input image (a) is split into its three color channels (b). For each color channel, 4 different pairs of pixels are used to generate 2-dimensional histograms (c). Horizontal, vertical, diagonal, and anti-diagonal pairs are considered. These histograms are then stacked to produce a single tensor (d). For some tests, only a subset of the co-occurrence matrices will be used.

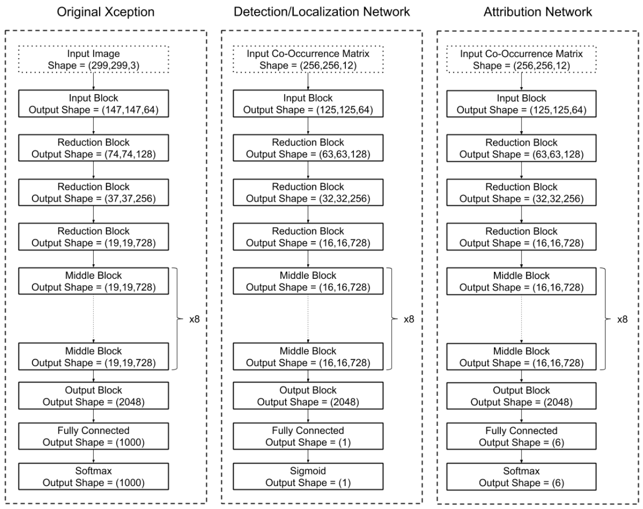


Figure 3: The original Xception network [15], shown next to our two modified models. Our architectures for detection and attribution are the same, except for the last layer and activation.

Convolutional Neural Networks

While the co-occurrence matrices are not themselves images, treating them as so has some theoretical backing. One of the primary motivations for using CNNs in image processing is their translation invariance property. In the case of a co-occurrence matrix, a translation along the main diagonal corresponds to adding a constant value to the image. We would not expect this manipulation to affect the forensic properties.

In this paper, we use Xception Net [15] deep neural network architecture for detection, attribution and localization of GAN generated images. The Xception network is a modified version of Inception network [75] but was created under a stronger theoretical assumption than the original Inception, where cross-channel correlations are completely split from spatial correlations by use depth-wise separable convolutions. The network also includes residual connections, as shown in Figure 3. For these reasons, the authors claim that Xception can more easily find a better convergence point than most other CNN architectures, while keeping model capacity low [15]. In this paper, we modify the original input and output shapes in the Xception network to accommodate our task as shown in Figure 3. The initial convolutional portions of the network remain unchanged, though the output sizes of each block are slightly different. This small change in size is accommodated by the global pooling step. Finally, the last fully connected layer of each network is changed to the desired number of output classes, and given the appropriate activation. For detection and attribution, our architectures are the same except for the last layer and activation. For localization, no changes were made to the model architecture but co-occurrence matrices were extracted on small image patches, and individually passed through the network.

Datasets

We evaluated our method on five different GAN architectures, of which each was trained on several different image generation tasks: ProGAN [37], StarGAN [14], CycleGAN [92], StyleGAN [38], and SPADE/GauGAN [65]. The modifications included image-to-image translation, facial attribute modification, style transfer, and pixel-wise semantic label to image generation. A total of 1.69 million real and 1.07 GAN generated images were In several cases, one or more images in the GAN generated category will be directly associated with an image in the authentic class. For example, a person's headshot untampered, blond, aged, and gender reversed will all be in the dataset. However, the splitting for training accounts for this, and will keep all of these images together to be put into either training, validation, or test.

StarGAN: This dataset consists of only celebrity photographs from the CelebA dataset [50], and their GAN generated counterparts [14]. The GAN changes attributes of the person to give them black hair, brown hair, blond hair, different gender, different age, different hair and gender, different hair and age, different gender and age, or different hair, age, and gender. These are the smallest of all of the training images, being a square of size 128 pixels.

CycleGAN: This datasets includes image-to-image translations between a wide array of image classes [92]. The sets horse2zebra, apple2orange, and summer2winter do a strict image-to-image translation, with the assumption that the GAN will learn the areas to modify. While the whole output is generated by the GAN, the changes for these will ideally be more localized. Ukiyoe, Vangogh, Cezanne, and Monet are four artists which the GAN attempts to learn a translation from photographs to their respective styles of painting. Facades and cityscapes represent the reverse of the image segmentation task. Given a segmentation map as input, they produce an image of a facade or cityscape. Map2sat takes in a Google Maps image containing road, building, and water outlines, and generates a hypothetical satellite image.

ProGAN: This dataset consists of images of celebrities, and their GAN generated counterparts, at a square size of 1024 pixels [37]. All data was obtained per the instructions provided in the paper's Github repository.

SPADE/GauGAN: SPADE/GauGAN contains realistic natural images generated using GANs [65]. This dataset uses images from ADE20k [90] dataset containing natural scenes and COCO-Stuff [12] dataset comprising day-to-day images of things and other stuff, along with their associated segmentation maps.

These untampered images are considered as real images in the GAN framework, and the pretrained models provided by the SPADE/GauGAN authors are used to generate GAN images from the segmentation maps.

StyleGAN: This dataset contains realistic images of persons, cars, cats and indoor scenes [38]. Images for this dataset were provided by the authors.

Experiments

Training Procedure

All deep learning experiments in this paper were done using Keras 2.2.5 and all training was done using an Adam optimizer [42], learning rate of 10^{-4}, and cross-entropy loss. A batch size of 64 was used for all experiments. Unless otherwise stated, a split of 90% training, 5% validation, and 5% test was used. Given the large amount of data available, a single iteration through the entire dataset for training took 10 hours on a single Titan RTX GPU. To allow for more frequent evaluation on the validation set, the length of an epoch was capped at 100 batches. Validation steps were also capped at 50 batches, and test sets at 2000 batches. After training for a sufficient period of time for the network to converge, the checkpoint which scored the highest in validation was chosen for testing. For experiments to determine hyper-parameters, training was capped at 50 epochs, and took approximately 3 hours each on a single Titan RTX. After determination of hyper-parameters, training of the final model was done for 200 epochs, taking approximately 12 hours.

Comparison with other CNN architectures:

First we evaluate our method on different well known CNN architectures: VGG16 [72], ResNet50 and ResNet101 [28],ResNet50V2, ResNet101V2 and ResNet152V2 [29], InceptionV3 and InceptionResNetV2 [75], and Xception [15]. Shown in Table 1 are the results for the different CNN networks. Though designed for ImageNet classification, all models take in an image with height, width, and 3 channels, and output a one-hot encoded label. The models are used as-is, with the following slight modifications. First, the number of input channels is set to be the depth of the co-occurrence feature tensor. Second, input shape was fixed at 256x256. Third, the number of output channels was set to 1. All of these parameters were passed as arguments to the respective Keras call for each model. A small margin separated the top performers, though Xception was the best with an accuracy of **0.9916** and had fewer parameters than others. For this reason, we chose Xception for the remainder of the experiments.

Comparison of Co-occurrence Matrix Pairs

Next we perform tests with different co-occurrence pairs, shown in Table 2. These experiments included JPEG compression, randomly selected from quality factors of 75, 85, 90, and no compression. Interestingly, it seems that the addition of more co-occurrence pairs did not significantly improve performance. For the remainder of the test, all 4 co-occurrence pairs were used.

Table 1: Comparison of different popular ImageNet [18] classification architectures on classifying GANs from co-occurrence matrices. All datasets are mixed for training, validation, and testing. The features are extracted from a whole image, with no JPEG compression.

Network	Accuracy
VGG16 [72]	0.6115
ResNet50 [28]	0.9677
ResNet101 [28]	0.9755
ResNet152V2 [29]	0.9795
ResNet50V2 [29]	0.9856
InceptionResNetV2 [75]	0.9885
InceptionV3 [76]	0.9894
ResNet101V2 [29]	0.9900
Xception [15]	**0.9916**

Table 2: Test on difference co-occurrence pairs. These were done on the whole image, with the additional challenge of JPEG compression. The JPEG quality factor was randomly selected with equal probability from the set of 75, 85, 90, or no JPEG compression

Pairs	Accuracy
Horizontal	95.51
Vertical	95.56
Hor and Ver	95.17
Hor, Ver, and Diag	**95.68**

Effect of patch size

For applications, the two parameters of interest were JPEG compression and patch size. The results for different patch sizes are shown in Table 3. These results are from images JPEG compressed by a factor randomly selected from 75, 85, 90, and none. A model is trained for each of the possible patch sizes, and then each model is tested against features from each patch size. It should be noted that in cases where the input image is smaller than the requested patch size, the whole image is used. There is notable generalization between different patch sizes, in that the model trained on a patch size of 256 and tested on 128 achieves an accuracy within a few percentage points of a model trained and tested on 128. Thus we would expect our models to work with a variety of untested patch sizes within a reasonable range while only taking a minor performance drop.

Effect of JPEG compression

Now assuming a fixed patch size of 128, we varied the JPEG quality factors: 75,85,90 and no compression. The model was again trained only on one particular JPEG factor as shown in Table 4. As expected, we see that performance increases with respect to quality factor. However, this table also shows that the model does not overfit to a particular quality factor, in that testing on a slightly better or worse quality factor gives a score not far from a model tuned to the particular test quality factor.

Table 3: Accuracy when trained on one patch size, and tested on another. Data for training and testing has been pre-processed using JPEG compression with quality factors randomly selected from 75, 85, 90 or none.

		Train		
		64	128	256
Test	64	0.7814	0.7555	0.6778
	128	0.8273	0.8336	0.8158
	256	0.8311	0.8546	0.8922

Table 4: Test accuracy when model is trained on images preprocessed with one JPEG quality factor, and tested on another.

		Train			
		75	85	90	None
Test	75	0.7738	0.7448	0.7101	0.6605
	85	0.8209	0.8593	0.8362	0.7209
	90	0.8310	0.8690	0.8756	0.7651
	None	0.9198	0.9386	0.9416	0.9702

Table 5: Train on all but one GAN, test on the held out images. Patch size of 128, no JPEG compression.

Test GAN	Accuracy
StarGAN	0.8490
CycleGAN	0.7411
ProGAN	0.6768
SPADE	0.9874
StyleGAN	0.8265

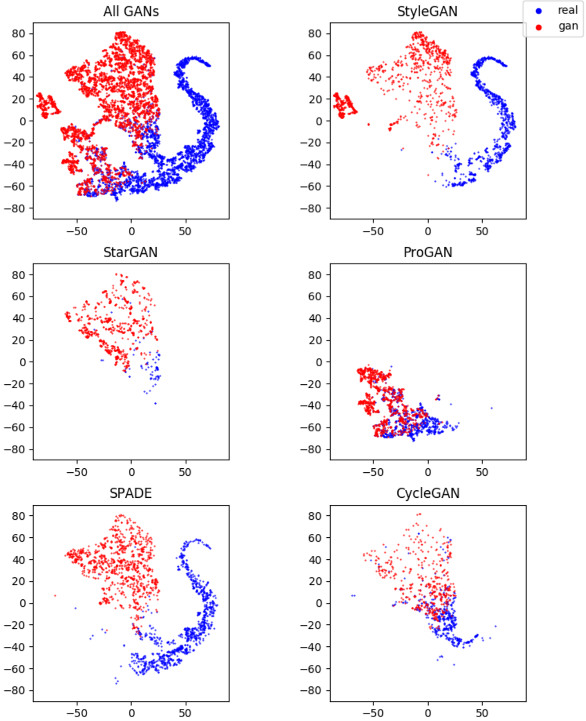


Figure 4: Visualization of images from different GAN datasets using t-SNE [52].

Generalization

To test the generalization between GANs, leave-one-out cross validation was used for each GAN architecture. One dataset of GAN images is used for testing and remaining GAN image datasets are used for training. Here, a patch size of 128 was used with no JPEG compression. From Table 5, we see that some GAN datasets such as SPADE, StarGAN and StyleGAN have high accuracy and are more generalizable. However, the accuracies for CycleGAN and ProGAN are lower in comparison, thus suggesting that images from these GAN categories should not be discarded when building a bigger GAN detection framework.

Visualization using t-SNE: To further investigate the variability in the GAN detection accuracies under the leave-one-out setting, we use t-SNE visualization [52] from outputs of the penultimate layer of the CNN, using images from the test set (as shown in Figure 4). The t-SNE algorithm aims to reduce dimensionality of a set of vectors while preserving relative distances as closely as possible. While there are many solutions to this problem for different distance metrics and optimization methods, KL divergence on the Student-t distribution used in t-SNE has shown the most promising results on real-world data [52].

To limit computation time, no more than 1000 images were used for a particular GAN from either the authentic or GAN classes. As recommended in the original t-SNE publication, the vector was first reduced using Principle Component Analysis (PCA). The original 2048 were reduced to 50 using PCA, and passed to the t-SNE algorithm. As we see in Figure 4, the images in CycleGAN and ProGAN are more tightly clustered, thus making them difficult to distinguish between real and GAN generated images, while the images from StarGAN, SPADE and StyleGAN are more separable, thus resulting in higher accuracies in the leave-one-out experiment.

Comparison with State-of-the-art

We compare our proposed approach with various state-of-the-art methods [56, 62, 89] on the CycleGAN dataset. In [56], Marra et al. proposed the leave-one-category-out benchmark test to see how well their methods work when one category from the CycleGAN dataset is kept for testing and remaining are kept for training. The methods they evaluated are based on steganalysis, generic image manipulations, detection of computer graphics, a GAN discriminator used in the CycleGAN paper, and generic deep learning architecture pretrained on ImageNet [18], but fine tuned to the CycleGAN dataset. Among these the top preforming ones were from steganalysis [23, 16] based on extracting features from high-pass residual images, a deep neural network designed to extract residual features [17] (Cozzolino2017) and XceptionNet [15] deep neural network trained on ImageNet but fine-tuned to this dataset. Apart from Marra et al. [56], we also compare our method with approaches including Nataraj et al. (Nataraj2019) [62], which uses co-occurrence matrices computed in the horizontal direction, and Zhang et al.(Zhang2019) [89], which uses spectra of up-sampling artifacts used in the GAN generating procedure to classify GAN images.

Table 6 summarizes the results of our proposed approach against other state-of-the-art approaches. Our method obtained the best average accuracy of **0.9817**, when compared with other methods. Even on individual categories, our method obtained more than 0.90 on all categories.

Tackling newer challenges like StyleGAN2

Apart from generalization, we tested our method on 100,000 images from the recently released StyleGAN2 [39] dataset of celebrity faces. The quality of these images were much better than the previous version and appeared realistic. When we tested on this dataset without any fine-tuning, we obtained an accuracy of 0.9464. This shows that our approach is promising in adapting to newer challenges. We also fine-tuned to this dataset by adding 100,000 authentic images randomly chosen from different GAN datasets, thus our new dataset comprised of 100,000 authentic images and 100,000 StyleGAN2 images. Then, we split this data into 40% training, 10% validation and 50% testing. When we

Table 6: Comparison with State-of-the-art.

Method	ap2or	ho2zeb	wint2sum	citysc.	facades	map2sat	Ukiyoe	Van Gogh	Cezanne	Monet	Average
Steganalysis feat.	0.9893	0.9844	0.6623	1.0000	0.9738	0.8809	0.9793	0.9973	0.9983	0.9852	0.9440
Cozzalino2017	0.9990	0.9998	0.6122	0.9992	0.9725	0.9959	1.0000	0.9993	1.0000	0.9916	0.9507
XceptionNet	0.9591	0.9916	0.7674	1.0000	0.9856	0.7679	1.0000	0.9993	1.0000	0.9510	0.9449
Nataraj2019	0.9978	0.9975	0.9972	0.9200	0.8063	0.9751	0.9963	1.0000	0.9963	0.9916	0.9784
Zhang2019	0.9830	0.9840	0.9990	1.0000	1.0000	0.7860	0.9990	0.9750	0.9920	0.9970	0.9720
Proposed approach	0.9982	0.9979	0.9982	0.9366	0.9498	0.9776	0.9973	0.9980	0.9993	0.9697	**0.9817**

Table 7: Number of images per class

	Train	Val	Test
Authentic	1,612,202	42,382	42,397
StarGAN	28,062	738	711
CycleGAN	17,265	439	439
ProGAN	70,286	1833	1,881
SPADE	138,075	3,717	3,704
StyleGAN	766,045	20,220	20,158

trained a new network on this dataset, we obtained a validation accuracy of 0.9984 and testing accuracy of 0.9972, thus also confirming that our approach can be made adjustable to newer GAN datasets.

GAN Attribution/Classification

While the primary area of interest is in determining the authenticity of an image, an immediate extension would be to determine which GAN was used. Here we perform an additional experiment on GAN class classification/attribution as a 6-class classification problem, the classes being: Real, StarGAN, CycleGAN, ProGAN, SPADE/GauGAN and StyleGAN. The number of output layers in the CNN was changed from 1 to 6, and output with the largest value was selected as the estimate. A breakdown of the number of images per class for training, validation and testing is given in Table 7. First, the network was trained where the input co-occurrence matrices were computed on the whole image. The training procedure was kept the same as with all other tests in the paper, with the exception of using a batch size of 60, and 10 images from each class per batch. This encouraged the network to not develop a bias towards any particular GAN for which we have more training data. First we consider the images as they are provided in the datasets. The classification results are shown in the form of confusion matrices in Table 8. For convenience, we also report the equal prior accuracy, equal to the average along the diagonal of the confusion matrix. This equal prior accuracy can be interpreted as the classification accuracy if each class is equally likely. We obtain an overall classification accuracy (considering equal priors) of 0.9654. High classification accuracy was obtained for most categories. StyleGAN had comparatively lower accuracy but still more than 90%, being mostly confused with SPADE/GauGAN and CycleGAN. These results show that our approach can also be used to identify which category of GAN was used.

Next, we trained the network using a patch size of 128×128 as input, and repeated the experiment. This is to see how well our method can be used for detection, localization as well as classification. The classification results are shown in Table 9. Now, we obtain an overall classification accuracy (considering equal priors) of 0.8477 (a drop of 12% when compared to full image accuracy). High classification accuracy was obtained for StarGAN, CycleGAN and ProGAN, while SPADE/GauGAN and StyleGAN had comparatively lower accuracies. These could be due to many factors such as the number of test images per class, patch size, and the authentic image datasets that were used for training in generating these GAN images.

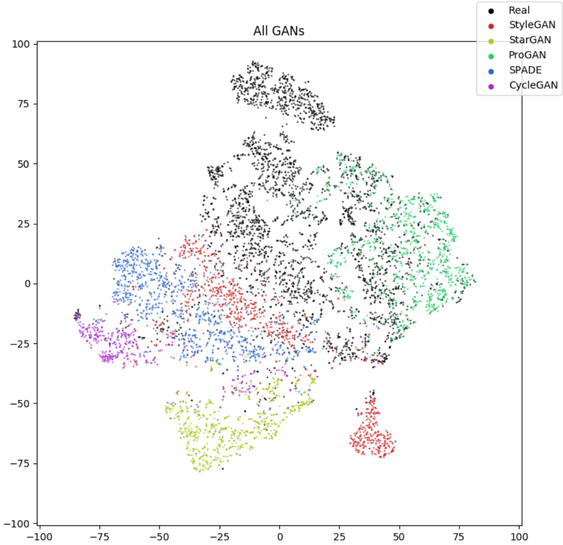


Figure 5: t-SNE visualization of 6 classes: Real, StyleGAN, StarGAN, ProGAN, SPADE/GauGAN and CycleGAN

In Table 10 we repeat the same experiment (with patch size 128×128) but with images that were randomly preprocessed with JPEG quality factors of 75, 85, 90, or no JPEG compression, with each of the four preprocessing methods equally likely. For this experiment, the overall classification accuracy drops slightly to 0.8088 due to the impact of JPEG compression.

For the multi-class experiment trained without JPEG compression, we repeat the t-SNE visualization procedure. Figure 5 shows all data-points on a single plot. These visualizations further support the results from the classification experiment.

Localization

Figure 6 show two example localization outputs. The image is processed in overlapping patches, with a particular stride and patch size. A co-occurrence matrix is then extracted for each patch, and passed through the CNN to produce a score. For pixels which are a part of multiple patches, the scores are simply the mean of all of the patch responses. These two examples use a patch size of 128, and a stride of 8. We can see that the heatmaps are predominantly blue for real images and predominantly red for GAN generated images. This further supports that our method

Figure 6: Localization heatmaps of (a) Real images and (b) GAN images from different GAN datasets (top to bottom): ProGAN [37], StarGAN [14], CycleGAN [92], StyleGAN [38], and SPADE/GauGAN [65].

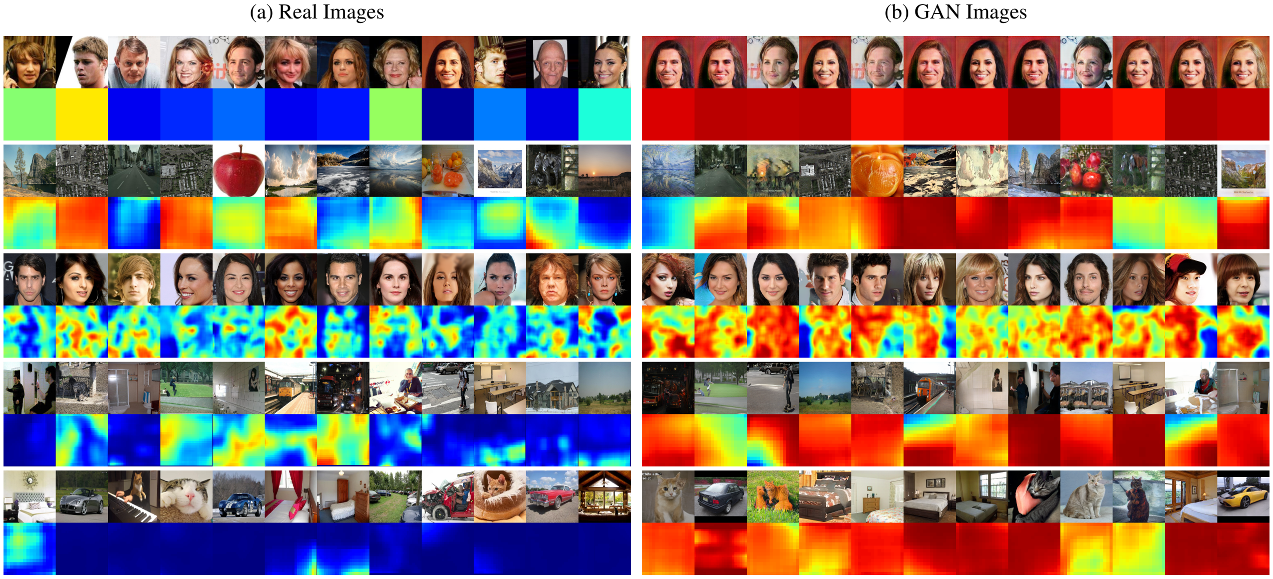


Table 8: Confusion matrix on images from GAN datasets without any pre-processing on the full image. Equal prior accuracy of 0.9654.

		Predicted Label					
		Real	StarGAN	CycleGAN	ProGAN	SPADE	StyleGAN
GT Label	Real	0.975	0.000	0.000	0.016	0.002	0.006
	StarGAN	0.000	0.976	0.014	0.000	0.010	0.000
	CycleGAN	0.000	0.000	0.964	0.000	0.036	0.000
	ProGAN	0.000	0.000	0.000	1.000	0.000	0.000
	SPADE	0.001	0.000	0.019	0.000	0.975	0.005
	StyleGAN	0.007	0.000	0.022	0.000	0.068	0.902

can be effectively used for GAN localization.

Conclusions

In this paper, we proposed a novel method to detect and attribute GAN generated images, and localize the area of manipulations. Detailed experimental results using a collection of over 2.7 million GAN and authentic images encompassing 5 major GAN datasets demonstrate that the proposed model is effective on a range of image scales and JPEG compression factors. In addition, the t-SNE visualization with our network's deep features showed promising separation of GAN and authentic images.

References

[1] Facebook removes bogus accounts that used ai to create fake profile pictures. https://www.cnet.com/news/facebook-removed-fake-accounts-that-used-ai-to-create-fake-profile-pictures/.

[2] Patent filings for generative ai have grown 500% this year as brands test its potential. https://www.adweek.com/digital/patent-filings-for-generative-ai-have-grown-500-this-year-as-brands-test-its-potential/.

[3] Removing coordinated inauthentic behavior from georgia, vietnam and the us. https://about.fb.com/news/2019/12/removing-coordinated-inauthentic-behavior-from-georgia-vietnam-and-the-us/.

[4] M. Albright, S. McCloskey, and A. Honeywell. Source generator attribution via inversion. *arXiv preprint arXiv:1905.02259*, 2019.

[5] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

[6] J. H. Bappy, A. K. Roy-Chowdhury, J. Bunk, L. Nataraj, and B. Manjunath. Exploiting spatial structure for localizing manipulated image regions. In *Proceedings of the IEEE International Conference on Computer Vision*, 2017.

[7] B. Bayar and M. C. Stamm. A deep learning approach to universal image manipulation detection using a new convolutional layer. In *Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security*, pages 5–10, 2016.

[8] B. Bayar and M. C. Stamm. Design principles of convolutional neural networks for multimedia forensics. In *The 2017 IS&T International Symposium on Electronic Imaging: Media Watermarking, Security, and Forensics*. IS&T Electronic Imaging, 2017.

[9] G. K. Birajdar and V. H. Mankar. Digital image forgery detection using passive techniques: A survey. *Digital Investigation*, 10(3):226–245, 2013.

[10] N. Bonettini, P. Bestagini, S. Milani, and S. Tubaro. On the use of benford's law to detect gan-generated images. *arXiv preprint arXiv:2004.07682*, 2020.

[11] J. Bunk, J. H. Bappy, T. M. Mohammed, L. Nataraj, A. Flenner, B. Manjunath, S. Chandrasekaran, A. K. Roy-Chowdhury, and L. Peterson. Detection and localization of image forgeries using resampling features and deep learning. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2017 IEEE Conference on*, pages 1881–1889. IEEE, 2017.

[12] H. Caesar, J. Uijlings, and V. Ferrari. Coco-stuff: Thing and stuff

Table 9: Confusion matrix on images from GAN datasets without any pre-processing on 128×128 patches. Equal prior accuracy of 0.8477.

		Predicted Label					
		Real	StarGAN	CycleGAN	ProGAN	SPADE	StyleGAN
GT Label	Real	0.826	0.003	0.016	0.021	0.066	0.068
	StarGAN	0.000	0.933	0.054	0.000	0.006	0.006
	CycleGAN	0.000	0.002	0.959	0.002	0.032	0.005
	ProGAN	0.000	0.002	0.008	0.981	0.004	0.005
	SPADE	0.001	0.025	0.210	0.008	0.728	0.029
	StyleGAN	0.003	0.025	0.101	0.009	0.203	0.659

Table 10: Confusion matrix with JPEG compression (128×128 patches). Equal prior accuracy of 0.8088. The images were preprocessed using a JPEG factor of 75, 85, 90, or no compression. Each of these four possible preprocessing functions was randomly selected with equal probability for every image.

		Predicted Label					
		Real	StarGAN	CycleGAN	ProGAN	SPADE	StyleGAN
GT Label	Real	0.741	0.005	0.020	0.026	0.103	0.104
	StarGAN	0.006	0.927	0.023	0.000	0.031	0.012
	CycleGAN	0.009	0.014	0.892	0.007	0.074	0.005
	ProGAN	0.002	0.003	0.009	0.973	0.007	0.007
	SPADE	0.075	0.015	0.095	0.009	0.765	0.042
	StyleGAN	0.114	0.021	0.059	0.008	0.243	0.555

classes in context. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1209–1218, 2018.

[13] Z. Chen and H. Yang. Manipulated face detector: Joint spatial and frequency domain attention network. *arXiv preprint arXiv:2005.02958*, 2020.

[14] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8789–8797, 2018.

[15] F. Chollet. Xception: Deep learning with depthwise separable convolutions. *arXiv preprint*, pages 1610–02357, 2017.

[16] D. Cozzolino, D. Gragnaniello, and L. Verdoliva. Image forgery detection through residual-based local descriptors and block-matching. In *Image Processing (ICIP), 2014 IEEE International Conference on*, pages 5297–5301. IEEE, 2014.

[17] D. Cozzolino, G. Poggi, and L. Verdoliva. Recasting residual-based local descriptors as convolutional neural networks: an application to image forgery detection. In *Proceedings of the 5th ACM Workshop on Information Hiding and Multimedia Security*, pages 159–164. ACM, 2017.

[18] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[19] A. E. Dirik, S. Bayram, H. T. Sencar, and N. Memon. New features to identify computer generated images. In *Image Processing, 2007. ICIP 2007. IEEE International Conference on*, volume 4, pages IV–433. IEEE, 2007.

[20] N.-T. Do, I.-S. Na, and S.-H. Kim. Forensics face detection from gans using convolutional neural network. In *ISITC'2018*, 2018.

[21] H. Farid. Image forgery detection. *IEEE Signal processing magazine*, 26(2):16–25, 2009.

[22] J. Frank, T. Eisenhofer, L. Schönherr, A. Fischer, D. Kolossa, and T. Holz. Leveraging frequency analysis for deep fake image recognition. *arXiv preprint arXiv:2003.08685*, 2020.

[23] J. Fridrich and J. Kodovsky. Rich models for steganalysis of digital images. *IEEE Transactions on Information Forensics and Security*, 7(3):868–882, 2012.

[24] 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.

[25] L. Guarnera, O. Giudice, and S. Battiato. Fighting deepfake by exposing the convolutional traces on images. *IEEE Access*, 8:165085–165098, 2020.

[26] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville. Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems*, pages 5767–5777, 2017.

[27] Z. Guo, G. Yang, J. Chen, and X. Sun. Fake face detection via adaptive residuals extraction network. *arXiv preprint arXiv:2005.04945*, 2020.

[28] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[29] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer, 2016.

[30] P. He, H. Li, and H. Wang. Detection of fake images via the ensemble of deep representations from multi color spaces. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 2299–2303. IEEE, 2019.

[31] 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.

[32] N. Hulzebosch, S. Ibrahimi, and M. Worring. Detecting cnn-generated facial images in real-world scenarios. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 642–643, 2020.

[33] S. Iizuka, E. Simo-Serra, and H. Ishikawa. Globally and locally consistent image completion. *ACM Transactions on Graphics (TOG)*, 36(4):107, 2017.

[34] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1125–1134, 2017.

[35] A. Jain, P. Majumdar, R. Singh, and M. Vatsa. Detecting gans and

retouching based digital alterations via dad-hcnn. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 672–673, 2020.

[36] A. Jain, R. Singh, and M. Vatsa. On detecting gans and retouching based synthetic alterations. In *9th International Conference on Biometrics: Theory, Applications and Systems*, 2018.

[37] 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.

[38] 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.

[39] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila. Analyzing and improving the image quality of stylegan. *arXiv preprint arXiv:1912.04958*, 2019.

[40] J. Kim, S.-A. Hong, and H. Kim. A stylegan image detection model based on convolutional neural network. *Journal of Korea Multimedia Society*, 22(12):1447–1456, 2019.

[41] T. Kim, M. Cha, H. Kim, J. K. Lee, and J. Kim. Learning to discover cross-domain relations with generative adversarial networks. In *International Conference on Machine Learning*, pages 1857–1865, 2017.

[42] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[43] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[44] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4681–4690, 2017.

[45] H. Li, H. Chen, B. Li, and S. Tan. Can forensic detectors identify gan generated images? In *APSIPA Annual Summit and Conference 2018*, 2018.

[46] 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.

[47] Y. Li, S. Liu, J. Yang, and M.-H. Yang. Generative face completion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3911–3919, 2017.

[48] X. Lin, J.-H. Li, S.-L. Wang, F. Cheng, X.-S. Huang, et al. Recent advances in passive digital image security forensics: A brief review. *Engineering*, 2018.

[49] M.-Y. Liu and O. Tuzel. Coupled generative adversarial networks. In *Advances in neural information processing systems*, pages 469–477, 2016.

[50] 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.

[51] Z. Liu, X. Qi, and P. H. Torr. Global texture enhancement for fake face detection in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8060–8069, 2020.

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

[53] B. Mader, M. S. Banks, and H. Farid. Identifying computer-generated portraits: The importance of training and incentives. *Perception*, 46(9):1062–1076, 2017.

[54] B. Mahdian and S. Saic. A bibliography on blind methods for identifying image forgery. *Signal Processing: Image Communication*, 25(6):389–399, 2010.

[55] H. Mansourifar and W. Shi. One-shot gan generated fake face detection. *arXiv preprint arXiv:2003.12244*, 2020.

[56] 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.

[57] F. Marra, D. Gragnaniello, L. Verdoliva, and G. Poggi. Do gans leave artificial fingerprints? *arXiv preprint arXiv:1812.11842*, 2018.

[58] F. Marra, C. Saltori, G. Boato, and L. Verdoliva. Incremental learning for the detection and classification of gan-generated images. *arXiv preprint arXiv:1910.01568*, 2019.

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

[60] M. Mirza and S. Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[61] H. Mo, B. Chen, and W. Luo. Fake faces identification via convolutional neural network. In *Proceedings of the 6th ACM Workshop on Information Hiding and Multimedia Security*, pages 43–47. ACM, 2018.

[62] L. Nataraj, T. M. Mohammed, B. Manjunath, S. Chandrasekaran, A. Flenner, J. H. Bappy, and A. K. Roy-Chowdhury. Detecting gan generated fake images using co-occurrence matrices. In *Media Watermarking, Security, and Forensics*. IS&T International Symposium on Electronic Imaging, 2019.

[63] J. Neves, R. Tolosana, R. Vera-Rodriguez, V. Lopes, H. Proença, and J. Fierrez. Ganprintr: Improved fakes and evaluation of the state-of-the-art in face manipulation detection. *arXiv preprint arXiv:1911.05351*, 2019.

[64] J. C. Neves, R. Tolosana, R. Vera-Rodriguez, V. Lopes, and H. Proença. Real or fake? spoofing state-of-the-art face synthesis detection systems. *arXiv preprint arXiv:1911.05351*, 2019.

[65] T. Park, M.-Y. Liu, T.-C. Wang, and J.-Y. Zhu. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2337–2346, 2019.

[66] G. Perarnau, J. van de Weijer, B. Raducanu, and J. M. Álvarez. Invertible conditional gans for image editing. *arXiv preprint arXiv:1611.06355*, 2016.

[67] T. Pevnỳ, P. Bas, and J. Fridrich. Steganalysis by subtractive pixel adjacency matrix. *information Forensics and Security, IEEE Transactions on*, 5(2):215–224, 2010.

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

[69] N. Rahmouni, V. Nozick, J. Yamagishi, and I. Echizen. Distinguishing computer graphics from natural images using convolution neural networks. In *Information Forensics and Security (WIFS), 2017 IEEE Workshop on*, pages 1–6. IEEE, 2017.

[70] Y. Rao and J. Ni. A deep learning approach to detection of splicing and copy-move forgeries in images. In *Information Forensics and Security (WIFS), 2016 IEEE International Workshop on*, pages 1–6. IEEE, 2016.

[71] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, pages 2234–2242, 2016.

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

2014.

[73] K. Sullivan, U. Madhow, S. Chandrasekaran, and B. Manjunath. Steganalysis for markov cover data with applications to images. *IEEE Transactions on Information Forensics and Security*, 1(2):275–287, 2006.

[74] K. Sullivan, U. Madhow, S. Chandrasekaran, and B. S. Manjunath. Steganalysis of spread spectrum data hiding exploiting cover memory. In *Security, Steganography, and Watermarking of Multimedia Contents VII*, volume 5681, pages 38–47. International Society for Optics and Photonics, 2005.

[75] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[76] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[77] Y. Taigman, A. Polyak, and L. Wolf. Unsupervised cross-domain image generation. *arXiv preprint arXiv:1611.02200*, 2016.

[78] S. Tariq, S. Lee, H. Kim, Y. Shin, and S. S. Woo. Detecting both machine and human created fake face images in the wild. In *Proceedings of the 2nd International Workshop on Multimedia Privacy and Security*, pages 81–87. ACM, 2018.

[79] R. Tolosana, R. Vera-Rodriguez, J. Fierrez, A. Morales, and J. Ortega-Garcia. Deepfakes and beyond: A survey of face manipulation and fake detection. *arXiv preprint arXiv:2001.00179*, 2020.

[80] R. Valle, U. CNMAT, W. Cai, and A. Doshi. Tequilagan: How to easily identify gan samples. *arXiv preprint arXiv:1807.04919*, 2018.

[81] S. Walia and K. Kumar. Digital image forgery detection: a systematic scrutiny. *Australian Journal of Forensic Sciences*, pages 1–39, 2018.

[82] R. Wang, L. Ma, F. Juefei-Xu, X. Xie, J. Wang, and Y. Liu. Fakespotter: A simple baseline for spotting ai-synthesized fake faces. *arXiv preprint arXiv:1909.06122*, 2019.

[83] S.-Y. Wang, O. Wang, R. Zhang, A. Owens, and A. A. Efros. Cnn-generated images are surprisingly easy to spot... for now. *arXiv preprint arXiv:1912.11035*, 2019.

[84] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8798–8807, 2018.

[85] R. Wu, X. Li, and B. Yang. Identifying computer generated graphics via histogram features. In *Image Processing (ICIP), 2011 18th IEEE International Conference on*, pages 1933–1936. IEEE, 2011.

[86] Z. Yi, H. Zhang, P. Tan, and M. Gong. Dualgan: Unsupervised dual learning for image-to-image translation. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2868–2876. IEEE, 2017.

[87] N. Yu, L. Davis, and M. Fritz. Attributing fake images to gans: Analyzing fingerprints in generated images. *arXiv preprint arXiv:1811.08180*, 2018.

[88] K. Zhang, Y. Liang, J. Zhang, Z. Wang, and X. Li. No one can escape: A general approach to detect tampered and generated image. *IEEE Access*, 7:129494–129503, 2019.

[89] X. Zhang, S. Karaman, and S.-F. Chang. Detecting and simulating artifacts in gan fake images. *arXiv preprint arXiv:1907.06515*, 2019.

[90] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba. Scene parsing through ade20k dataset. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 633–641, 2017.

[91] P. Zhou, X. Han, V. I. Morariu, and L. S. Davis. Learning rich features for image manipulation detection. *arXiv preprint arXiv:1805.04953*, 2018.

[92] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *IEEE International Conference on Computer Vision*, 2017.

[93] Y.-X. Zhuang and C.-C. Hsu. Detecting generated image based on a coupled network with two-step pairwise learning. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 3212–3216. IEEE, 2019.

Acknowledgements

This research was developed with funding from the Defense Advanced Research Projects Agency (DARPA). The views, opinions and/or findings expressed are those of the author and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government. The paper is approved for public release, distribution unlimited.

Author Biography

Michael Goebel *received his B.S. and M.S. degrees in Electrical Engineering from Binghamton University in 2016 and 2017. He is currently a PhD student in Electrical Engineering at University of California Santa Barbara.*

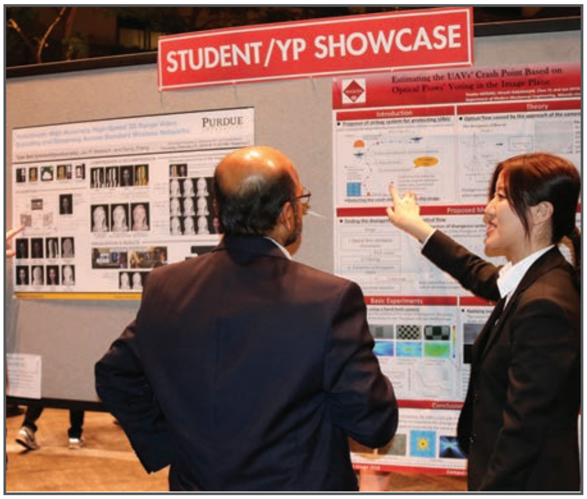
Lakshmanan Nataraj *received his B.E degree from Sri Venkateswara College of Engineering, Anna university in 2007, and the Ph.D. degree in the Electrical and Computer Engineering from the University of California, Santa Barbara in 2015. He is currently a Senior Research Staff Member at Mayachitra Inc., Santa Barbara, CA. His research interests include malware analysis and image forensics.*

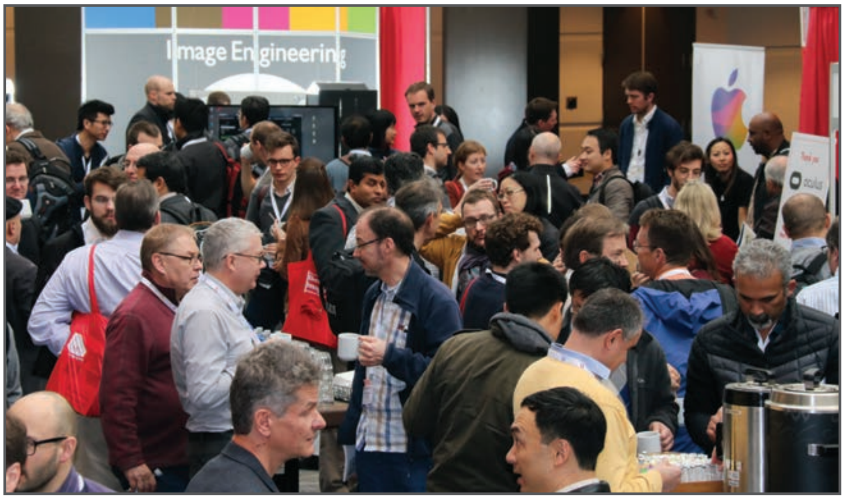
Tejaswi Nanjundaswamy *received his Ph.D. degree in Electrical and Computer Engineering from the University of California, Santa Barbara (UCSB), He is currently working in Apple where his research interests include audio and speech processing/coding. Mr. Nanjundaswamy is a student member of the Audio Engineering Society (AES). He won the Student Technical Paper Award at the AES 129th Convention.*

Tajuddin Manhar Mohammed *received his B.Tech (Hons.) degree from Indian Institute of Technology (IIT), Hyderabad, India in 2015 and his M.S. degree in Electrical and Computer Engineering from University of California Santa Barbara (UCSB), Santa Barbara, CA in 2016. After obtaining his Masters degree, he obtained a job as a Research Staff Member for Mayachitra Inc., Santa Barbara, CA. His recent research efforts include developing computer vision techniques for image forensics and cyber security.*

Shivkumar Chandrasekaran *received his Ph.D. degree in Computer Science from Yale University, New Haven, CT, in 1994. He is a Professor in the Electrical and Computer Engineering Department, University of California, Santa Barbara. His research interests are in Computational Mathematics*

B. S. Manjunath *(F'05) received the Ph.D. degree in Electrical Engineering from the University of Southern California in 1991. He is currently a Distinguished Professor at the ECE Department at the University of California at Santa Barbara. He has co-authored about 300 peer-reviewed articles. His current research interests include image processing, computer vision and biomedical image analysis.*





AR is a fundamental and transformative interface and platform.

20+ years from my initial working demo to today's AR displays

Optimistic that it won't take another 20 to reach ubiquitous consumer AR!