Computing and Informatics, Vol. 41, 2022, 1541–1562, doi: 10.31577/cai_2022_6_1541

EXPOSING AI GENERATED DEEPFAKE IMAGES USING SIAMESE NETWORK WITH TRIPLET LOSS

Sidra Kanwal, Samabia Tehsin, Shahela Saif

Department of Computer Science
Bahria University, Islamabad, Pakistan
e-mail: `01-243182-018@student.bahria.edu.pk`,
`stehseen.buic@bahria.edu.pk`,
`01-284181-002@student.bahria.edu.pk`

Abstract. Generative Adversarial Networks have gained popularity mainly due to their ability to create fake human faces. The remarkable detail with which such images have been created in the past few years has exceeded the ability of humans to differentiate between these fake images and real images. Such images have been known to be capable of deceiving the face recognition systems with certain success as well. Forensic systems being developed nowadays take into account adversarial attacks in order to create a more comprehensive detection approaches. Different GAN algorithms such as StackGAN, StyleGAN use different architectures to produce images. Since the underlying technique is different from one another it is difficult for any single detection algorithm trained on one kind of GAN to detect fake images generated from some other kind of GAN. In this research we use a siamese network with triplet loss function to provide a generic solution for detection of GAN generated images or deepfake images. Extensive experiments have been conducted to analyze the effectiveness of the proposed approach. The results show that the siamese triplet loss network performs significantly better than the contemporary approaches with accuracy exceeding 90 % in most experiments.

Keywords: Digital forensics, deepfake images, image forgery, siamese triplet loss, AI generated images

Mathematics Subject Classification 2010: 68T45, 68T10

1 INTRODUCTION

In the past few years we have seen a tremendous amount of growth in the field of Artificial Intelligence and Deep Learning, notable among these are the Generative Adversarial Networks. One truly remarkable outcome of GANs is the photo-realistic synthetic media including images and videos [1]. GANs have found their applications in diverse computer vision fields including conditional image translation [2], text-to-image synthesis [3], style mixing [4], cross domain context matching [5] and domain specific high quality image synthesis [6]. Creating falsified human faces in images or videos is commonly referred to as deepfakes.

As it can be seen from Figure 1, the visual quality of synthesized media (or deepfakes) produced by GANs is not discernible from real content without the aid of any specialized tools. This has numerous useful applications in multimedia field such as remaking old movies, creating realistic animations both of fictional characters and real people that are a part of history now. However, deepfakes are more frequently used for malicious purposes with ill intent. This creates many challenges that have widespread impact if left unchecked, for example – using deepfake images to create fake identities, creating deepfake videos for the purpose of exploitation and promoting false propaganda. Such applications create concerns in numerous fields including security systems reliability, political deception and misconceptions and legal data's authenticity [7, 8, 9]. One factor that makes the problem of deepfakes more urgent and dangerous is the use of social media to spread such falsified content – social media not only has a widespread effect but is also known for the incredibly fast speed at which any information spreads using it.

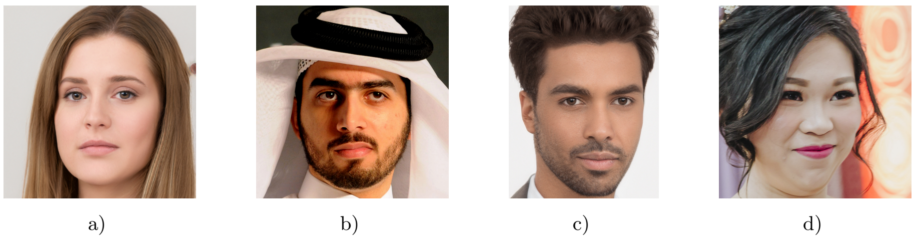

Figure 1. Visual comparison of images randomly taken from PGGAN dataset. a), c) are artificial and b), d) are real. GAN synthesized images are realistic and high-quality posing a unique challenge in their identification.

The magnitude of the threat posed by deepfakes makes it even more important that a solution is devised for its timely and effective detection. Visual media in most cases is considered irrefutable, and thus can have a much more lasting impact than any information given through other media such as text. Research done in order to detect these deepfake forgeries have enjoyed a degree of success [10, 11, 12, 13, 8, 9, 14], mostly owing to the fact that the deepfake content generation algorithms are not fully mature yet. The current focus is in detecting deepfake image or video forgery using statistical and/or morphological properties of images. These solutions, however, do not produce the same quality of results when tested on in-the-wild deepfakes or even deepfakes generated from a different GAN algorithm. Any effective solution will need to look to create a more sustainable solution by ensuring that it is robust to unseen or in-the-wild deepfakes.

Our work focuses only on deepfake images and provides a generalized solution compared to the state-of-the-art deepfake image detection solutions. This study highlights the shortcoming of the current solutions and addresses those by making use of a triplet loss based siamese network. Numerous experiments are conducted and results have shown that use of a triplet loss helps make the detection solution more robust and effective.

The paper is organized as follows. Section 2 discusses literature review related to the problem, while problem formulation is discussed in Section 3. Data preprocessing is described in Section 4. Experimental results and evaluation of the proposed research are provided in Section 5. Finally, the conclusion and future direction are stated in Section 6.

2 RELATED WORK

2.1 Generative Adversarial Networks (GANs)

GANs [1, 2, 15, 5, 6, 4, 16, 17, 3, 18, 19] have the ability to create photo-realistic visual media such as images and videos.

Multiple applications are freely available to the public [20, 21, 22] that can be used to generate the fabricated images, most of these images change or manipulate a person's identity or facial expressions. GANs were first introduced by Goodfellow et al. in [1] where a generative architecture was presented with the capability to generate the data given some similar examples as training data. GANs have improved tremendously since then.

Karras et al. in [6] and [4] have created GANs with high quality, conditionally styled, more consistent, and stable graphics with negligible artifacts. Choi et al. [2] have designed a unified GAN model for the image to image translation. Isola et al. [18] proposed a model that gives high-quality images that are translated into other domains using GANs. Another improvement in the domain of image-to-image translation is proposed by Zhu et al. [19], in which pair-to pair translation is done by using a pair of data examples, one example X is used as input and the other is used as Y or output image reference. In previous researches, some focused on the resolution of reconstructed images by comparing with the original image resolution [23]. Li and Wand [24] improve the GAN generated images' quality in real-time, in which the model generates 512×512 images within 40 ms. In [25] authors use the unsupervised mechanism for generating the synthetic images using the joint distribution of input images of different domains. GAN models are converted into the recurrent neural network by Im et al. [15], where a generator, and discriminator both compete with each other in the training process. Zhao et al. emphasize on energy factor in adversarial networks to provide the minimal energy-based synthetic images [16].

2.2 Fake Face Forensics

There are numerous studies on the detection of AI-generated fake photos or videos using deep learning models, which can be categorized into two broad segments: deepfake video forensics and deepfake image forensics.

2.2.1 Deepfake Video Detection

Deepfake videos have recently become an important discussion topic due to their impact in causing discord and chaos in political and religious situations. They are commonly used to disgrace politicians and celebrities. Video forensics [8, 26, 27, 28, 11, 29] has thus been an active area of research. Güera and Delp [8] present a two-stage method that extracts the features from frames using a CNN. These features are passed to a Recurrent Neural Network that differentiates the deepfake videos from real ones based on image inconsistencies. The shortcoming of this approach is that it fails to find the forgery artifacts in small regions. Li and Lyu [26] gave another method that can detect the deepfake videos by utilizing the concept of face wrapping artifacts, it performs well but these artifacts are not seen in all deepfake videos so the solution is not generalizable In some of the investigations [27, 28], researchers consider the morphological feature of frames like eye-blinking, head position, etc. for video forensics. Afchar et al. [11] demonstrate the failure of forgery detection methods based on morphological features in the case of compressed videos and presents the multi-layer solution based on CNN. Most of the forensics techniques proposed in literature do not work in low-resolution deepfake videos. Their effectiveness is also linked to the algorithm used for generation of deepfake media, which limits their success to certain scenarios.

2.2.2 Deepfake Image Detection

Image forenscis has been used to identify manipulated photos, enhanced photos, image-to-image translations and deepfake facial images [9], [12, 30, 31, 32]. Deepfake images have been used recently to spoof security systems that work on facial identity. These have also been employed to create fake identities on social media to spread fake news and to promote false propaganda. Deepfake face image forgeries can be created by either doing a faceswap or an expression swap. A faceswap replaces the facial features of target person's with those of source person, while expression swap modified the facial features of target person based on those of source person. Quite some work has been done to detect deepfakes [10, 14, 33, 13, 34]. Many of these techniques employ machine learning and deep learning for image forensics based on the pixel-based attributes.

Rössler et al. [12] devised the method for detection of FaceSwap, Face2Face, and deepfake manipulated images by using deep convolutional neural networks. Marra et al. [35] identify the image-to-image translation visual forgery and claim that fabricated, altered, tempered, or manipulative visual forgery is relatively easy to detect rather than GANs. Li et al. [14] in 2019 proposed a model based on pixel color dissimilarities like a flaw of color difference, distortion, and any other kind of artifact. Co-occurrence metrics are used to measure these artifacts from images which are classified using deep neural network. In [12], the authors focus on the specific fabrication of media, e.g. face swap, whereas another study [13] deals with the human crafted and GAN fake images forgery. In another research [10], the author proposed the techniques based on the neural architecture that can detect the specific type of forgery like FaceSwap, and emotion change.

Wang et al. [10] proposed a new method based on examining the neuron behavior. They tested their system on Style and PGAN fake images are identified in the study. The proposed method observes the neuron behavior layer by layer, gives a positive result for the fake image, and finally, fine-grained features are extracted and input to SVM classifier that classify the input image. As GANs become more powerful, we need more sophisticated detection methods as well. Xuan et al. [33] work on the generalization of forensics methods by employing the noisy fake images just to remove the basic artifact and enforce the model to learn more and more intrinsic feature that enhance the generalization of proposed model. Spectral input rather than pixel-wise has been used in [36]. Their technique used GANs artifacts created during the up-sampling of fake images as an indicator of deepfakes. There is limited literature that is specifically carried out for GANs forensics which identifies the region of tampering as well. Dang et al. [37] identified the GAN images by employing a CNN that works on the pre-processed dataset (only tampered or GAN region of the image) to recognize the fake image. On the other hand, Tariq et al. [13] devised the classifier for human created and Progressive GAN images and have also devised the end-to-end framework for automatic detection [34].

2.3 Vulnerabilities of Visual Forensics

Stat-of-the-art face recognition systems have a lot of vulnerabilities in terms of their robustness. Korshunov and Marcel [38] evaluate the vulnerabilities of current face recognition systems regarding image morphologies in 2019. Performance has been tested using two architectures: Facenet and VGG, with a False Acceptance Rate of 95 % and 85 %, respectively. In the survey state-of-the-art forensics techniques are also evaluated for deepfake video detection that are created using GANs. Research demonstrates that GAN generated visual contents are more challenging for face recognition systems, as well as existing detection system, and in the future, advanced technologies for deepfake creation will make it even more difficult to differentiate between a real image and a fake one [38]. Fake media content creates serious community issues as it takes time to find the credibility of visual content.

New tools and techniques are designed for the recognition of fake visual content but the synthetic media generation algorithms are also evolving.

Numerous studies exist for deepfake video forensics while few are available for image forensics. In limited research regarding image forensics, most methods do not focus on detection of GAN generated visual contents. GANs forensics techniques are designed with machine learning and deep neural networks. State-of-the-art methods show high positive prediction with less false results but the catch is the training data. All these models provide noteworthy results when the training and test dataset contain images generated using the same GAN technique, but show a significant drop in performance when the training and test dataset come from different (GAN) algorithms. Therefore state-of-the-art deepfake forensics' performance is yet to be seen in the real-world environment.

The proposed study is aimed at providing a more generic solution – we use training and test samples collected from different datasets and use a triplet loss to find feature embeddings that are less effected by the generation algorithm and hence perform better with unseen data.

2.4 Siamese Network

Siamese networks are deep networks that have two streams or more, where each stream shares the same learning weights. These are often refered to as one-shot learning architectures due to their applications in domains with limited data. Siamese networks use loss functions, such as contrastive loss or triplet loss, which ensure that not only is each example transformed into an equivalent feature embedding but also the feature embeddings of the same class examples lie closer together while feature embeddings of different class examples are far apart. Intuition behind the one-shot learning is to find the similarities between the same class and differences between the different class examples. Moreover, siamese triplet loss network is an advanced form of one-shot learning in which three samples are taken at one time and triplet loss is measured in each iteration for learning rather than simple cost function [39]. We have used siamese network because of this ability to create meaningful embeddings which will allow for robustness towards unseen data.

3 PROPOSED METHODOLOGY

In this section, a detailed discussion is provided on the proposed methodology and the architecture used in this study.

3.1 Data Preparation

Three datasets were used for the experiments carried out in this study, PGAN, FFHQ, and StyleGAN. Figure 2 shows the data preparation methodology.

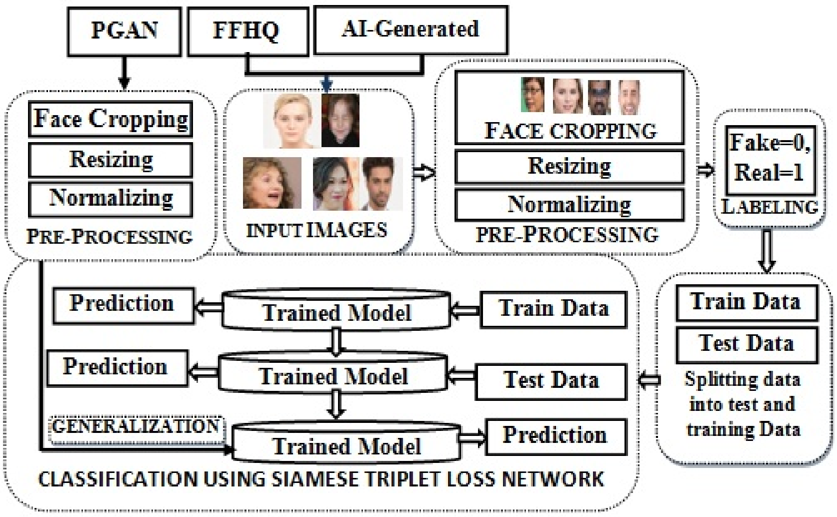

Figure 2. Proposed methodology comprises three data samples that are pre-processed with model training, evaluation, and generalization.

4 DATA PRE-PROCESSING

Since our main focus is on identification of fake faces from real ones, we pre-process the images to extract the facial area as is given in Algorithm 1. Samples I are selected, which are at the first step re-scaled to 128×128 sized images. Centering and rescaling of images to extract face area improves the accuracy of the algorithm. The resized images are normalized using the *Z-score* calculated as given by the formula:

$$Z_i = \frac{x_i - \mu}{\sigma}. \tag{1}$$

In Equation (1), x_i is the given sample, μ is the mean, and σ shows the standard deviation of samples. Applying standard normalization, value of μ is set to 0, and σ to 1. Normalization improves the overall efficiency of the algorithm. The categorical labels are converted to numeric labels, where 0 represents the fake or GAN generated image and 1 represents the real image.

4.1 Classification

A typical neural network learns the similarities between the input samples to classify them in the given categories whereas a CNN learns the features and the similarities for classification. These networks perform best when the training samples are similar to the test samples. A set of experiments were conducted to support this argument, as presented in Table 2 where we can see that the pretrained networks provide

```
**Input**: $I$ = Face Image
**Parameter**: rescale = 1.05, min_n = 6
**Output**: $F$ = Cropped Face
$I$ = Browse Image
grey = Greyscale conversion
faces = detect_multi_scale(grey, rescale, min_n)
if faces is then
    while $(x, y, h, w)$ in faces do
        face = crop($I$, $(x, y, h, w)$)
        $F$ = resize(face, $[128, 128]$)
    end
else
    go back to the image browse;
end
return $F$
```

Algorithm 1: Algorithm for face cropping and resizing

an accuracy of over 90 % when the dataset used for training is similar to the one used for testing. However, the problems in real world seldom have such a perfect balance. A siamese network finds the distance between the similarities found between samples of the same class and those of different classes, enabling us to work on smaller data just as successfully as on larger data [39]. This is why it is often used for one-shot learning or classification. A siamese network uses three inputs: anchor sample A, the positive sample P and the negative sample N. A triplet loss is calculated that minimizes the distance of the positive sample to the anchor, while maximizing the distance of the negative sample to the anchor.

The siamese network in our study uses five convolutional layers followed by a fully connected layer. Figure 3 represents the proposed siamese architecture. The initial layers take as input an image of $128 \times 128 \times 3$ dimension that is reduced to $27 \times 27 \times 128$ before being passed to the final classification layer. The network consists of two blocks each comprising of two convolutional layers followed by a single max pooling layer. Both layers in first block use 32 filters while in second use 64 filters. The final convolutional layer uses 128 filters, followed by flattening and classification layers. The network uses shared weights while creating embedding for each image in the sample, values of which are later used for calculating loss.

The network seeds with Xavier weights that gives normalized values for initialization that are normally distributed with mean zero and the standard deviation is $\frac{1}{N}$, where N is the number of input neurons. For activation of the hidden layer, ReLU is used that gives $\max(x, 0)$ and no activation at the output layer as the proposed solution finds the embedding vector rather than classification. Furthermore,

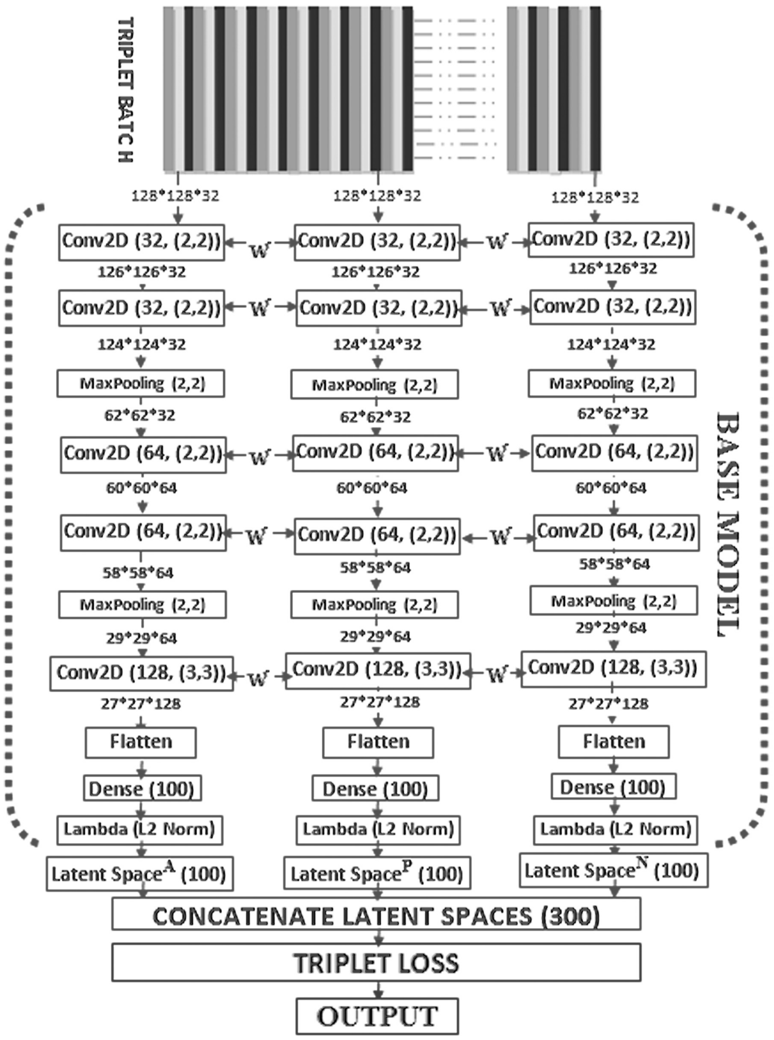

Figure 3. 12-layer proposed architecture with input shape 128 $\times$ 128, and output 100-dimensional embedding vector. Two conv2D preceding with max pooling, extract the desired feature that are flatten, normalize, and concatenate to measure the cost for model learning.

lambda layer transforms the values into l_2 norm form, given in Equation (2). For input samples $x = x_1, x_2, \ldots, x_d$ with dimension d, l_2 norm will be:

$$||X|| = \sqrt{\sum x_i^2} = \sqrt{x^T \cdot x}. \tag{2}$$

```
Input: Predicted Embedding vector
Output: loss= Calculated loss
 1: $anchor\_out \longleftarrow embedding_A$
 2: $positive\_out \longleftarrow embedding_P$
 3: $negative\_out \longleftarrow embedding_N$
 4: $positive\_distance \longleftarrow Dist(anchor\_out, positive\_out),$
 5: $negative\_distance \longleftarrow Dist(anchor\_out, negative\_out)$
 6: $basic\_loss \longleftarrow Diff(positive\_distance, negative\_distance)$
 7: $exponent\_loss \longleftarrow 1 + \exp(basic\_loss)$
 8: $loss \longleftarrow \log(exponent\_loss)$
    return $loss$
```

Algorithm 2: Algorithm for triplet loss

The triplet loss function uses latent space vectors of A, P, N and labels of samples as input parameters. These vectors are assigned to the corresponding variables, distances *positive_distance* between A, P and *negative_distance* between A, N are calculated in steps 4, 5. Their difference *basic_loss* is measured in step 6 that is utilized for further steps. Exp is applied on *basic_loss* in step 7 followed by log of *exponent_loss* which is the final *loss*. This loss *loss* is back propagated before the next iteration. To decide the weightage of loss or learning rate that will be propagated in each iteration, Adam optimizer is used. As compared to Adagrade and RMSprop that consider only the average of the previous gradient, Adam calculates first- and second-order derivatives of the moment of the gradient [40].

$$m_t = \frac{m_t}{1-\beta_1^t},$$

$$v_t = \frac{v_t}{1-\beta_2^t}, \tag{3}$$

$$\theta_{t+1} = \theta_1 - \frac{\mu}{\sqrt{v_t}+\epsilon} m_t. \tag{4}$$

Here, m_t, v_t show the first, and the second order of the moment calculated with the help of gradient decent values. In equation m_t, v_t are initialized with zero, β_1, β_2 are 0.999 close to one, ϵ is set as 10^{-8} and μ is learning rate. Equation (4) is used for update of weights, θ_t, θ_{t+1} show the previous and new weights consecutively.

4.1.1 System Requirements

The experiments were carried out on GTX 1080 Ti GPU with 32-bit, 25 GB RAM, 550 GB SSD, and (PSU) 1 000–1 500 W (power supply). Python 3.7.0 with TensorFlow and Keras were used.

5 RESULTS AND EVALUATION

In this section we first demonstrate the ability of pre-trained models such as ResNet and VGG to detect fake images while they are trained and tested on the same dataset. Later we evaluate them on different training and test datasets and introduce our own model for comparison. Firstly, we discuss the dataset distributions and the performance parameters.

5.1 Dataset Distributions

The experiments conducted are split into two cases:

Case I – where training and test sets come from the same dataset;

Case II – where training and test set come from different datasets.

For the first case, the real images are taken from FFHQ dataset, while fake samples are generated using StyleGAN. The distributions for training and test samples are given in Table 1. In the second case, we train our system on FFHQ and StyleGan while testing it using images generated by PGAN.

Dataset	Training	Testing	Total
Case I			
StyleGAN	30 k	10 k	40 k
FFHQ	30 k	10 k	40 k
			80 k
Case II			
StyleGAN	40 k		40 k
FFHQ	40 k		40 k
PGAN		40 k	40 k
			120 k

Table 1. The number of samples for training and testing sets on each sample class

5.2 Performance Measures

For a thorough investigation, we have used multiple performance measures including Accuracy, Precision, Recall, F1-score, Cohen Kappa, ROC, and PRC. Given ahead are the mathematical formulations for each one of these.

Accuracy is calculated as a ratio of true positives to the total samples we have, as given in Equation (5).

$$Accuracy = \frac{TP}{TP + TN + FP + FN} \tag{5}$$

where

- TP = True Positive (Set of examples that belongs to fake face class and predicted as fake),
- TN = True Negative (Set of examples that belongs to real face class and predicted as real),
- FP = False Positive (Set of examples that belongs to real face class and predicted as fake),
- FN = False Negatives (Set of examples that belongs to fake face class and predicted as real).

Precision (Equation (6)) and Recall (Equation (7)) are measures of True Positives compared to False Positives and False Negatives. While F1-Score (Equation (7)) is the ratio between Precision and Recall.

$$Precision = \frac{TP}{TP + FP}, \tag{6}$$

$$Recall = \frac{TP}{TP + FN}, \tag{7}$$

$$F1\text{-}Score = \frac{2 \times (Recall \times Precision)}{Recall + Precision}. \tag{8}$$

Cohen Kappa is given by:

$$\kappa = \frac{P_0 - P_e}{1 - P_e} \tag{9}$$

where P_0 is identical to accuracy, and P_e is the estimated probability calculated using the observed values of each category separately and it can be given by:

$$P_e = \frac{(TP + FP) \times (TP + FN) + (TN + FP) \times (TN + FN)}{(TPS) \times (TNS)}. \tag{10}$$

In Equation (10), TPS and TNS are the total positive and negative samples respectively used for evaluation of the model.

The receiver operating curve (ROC) and precision recall curve (PRC) is also used for the assessment of the proposed model. The model is considered as best as its area under the curve (AUC) converges to 1.

5.3 Proposed Model

A siamese network focuses not only on predicting a true label for given data but also tries to pull together examples belonging to the same class while pulling apart the examples from different classes as much as is possible. Contrastive loss and triplet loss are used for this purpose. We have used triplet loss for the calculation of which we need three inputs: a positive example, a negative example and an anchor. The algorithm minimizes the distance between the positive example and the anchor and maximizes the distance between the negative example and the anchor.

We created two variations of the siamese network. The first variation consists of a 12-layered architecture consisting of convolution layer, pooling layers, normalization layer and dense layers. The architecture has 3 streams corresponding to each of the input images. Each input has the same dimension of $128 \times 128 \times 3$ and each stream uses the same number and size of filters. This allows for weight sharing to be possible among all three streams as is an integral part of a siamese network. Figure 3 provides a representation of this architecture.

In the second variation to our siamese network, we have used ResNet50 as the base architecture. Each stream of the siamese network is one instance of the ResNet50. However, we see that in Case I shallow network as base performs better than ResNet50.

5.4 Case I Experiments

In Case I we extracted training and test samples from the same datasets. In order to detect deepfake images, we first use the transfer learning technique. In this study we use five pre-trained models: VGG16, VGG19, ResNet50, InceptionV3, and Xception net.

5.4.1 Experimental Setup

VGG16 is 16-layer architecture with 13 convolutional and 3 dense layers while VGG19 is a 19-layer in which 16 are convolutional and 3 dense layers. The network uses pretrained imagenet weights with $128 \times 128 \times 3$ input where we freeze the initial layers and retrain the dense layers. Both models converge in less than 35 epochs, with early stopping criteria based upon validation loss, and take no more than an hour of execution on the GPU.

InceptionV3 is 48-deep layer architecture with an 11-inception module and each module contains the convolutional, pooling, and ReLU activation layers [41]. Xception is another model that holds 36-convolutional layers which are arranged into 14 modules with the linear residual connection between them [42] and ResNet50 is a 50-layer model incorporated with 49 convolutional, max-pooling, and average pooling layers [43]. These models are also trained with imagenet weights, $128 \times 128 \times 3$ input shape, where we retrain all the layers instead of freezing the top layers. InceptionV3 achieves highest accuracy in 14 epochs with approximately 1 hour, Xception, ResNet50 converge in 18, 28 epochs with approximately 63 min., and 1 hour and 45 min., respectively.

The performance of each of these architectures is given in Table 2.

Model	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**Kappa**	**AUC**
VGG16	94.67 %	95.84 %	93.4 %	94.6 %	89.3 %	98.7 %
VGG19	94.98 %	95.53 %	94.4 %	94.9 %	89.9 %	98.9 %
ResNet50	98.93 %	98.93 %	98.9 %	98.9 %	97.9 %	99.9 %
InceptionV3	96.81 %	99.70 %	93.9 %	96.7 %	93.6 %	97.2 %
Xception	96.09 %	97.01 %	95.1 %	96.0 %	92.2 %	98.8 %
Proposed Model	94.8 %	94.6 %	94.9 %	94.8 %	96.9 %	94.9 %

Table 2. Comparison of pre-trained networks and proposed model on Case I where training and test samples are from the same dataset

5.4.2 Results

From experiments, it can be concluded that VGG16, VGG19 yield similar results, while InceptionV3 and Xception performs better than VGG16, VGG19 with 96.81 %, 96.09 % accuracy, respectively. Moreover, ResNet50 gives significantly better results in terms of performance metrics Accuracy, Precision, Recall, F1-score, Kappa, and AUC. ResNet has a powerful feature of skip connection that stops the model deterioration due to deep network. Our model also provides performance comparable to the pre-trained models even though it is not the-highest performance. Since ResNet50 outperformed all other pre-trained networks, therefore, we provided a closer look at its learning curves. Figure 4 gives the performance learning curve and Figure 5 shows the model optimization learning curve for ResNet50.

Figure 4 shows the 2-dimensional plotting of training and validation accuracies during each epoch. After initial fluctuations, the model starts to converge after 15th epoch. We see another performance degradation after epoch 28 where the model tends to overfit.

Figure 5 shows the loss curve of the ResNet50 model with validation and training loss. Much like model accuracy, the loss measure improves after the 15th epoch.

It can be seen that ResNet50 shows significant results with high number of correct predictions. The true positive rate (TPR) and false-positive rate (FPR) for ResNet50 are given below [44]:

$$TPR = \frac{TP}{TP + FN} = \frac{9\,951}{9\,951 + 49} = 0.9951, \tag{11}$$

$$FPR = \frac{FP}{FP + TN} = \frac{119}{119 + 9\,881} = 0.0119. \tag{12}$$

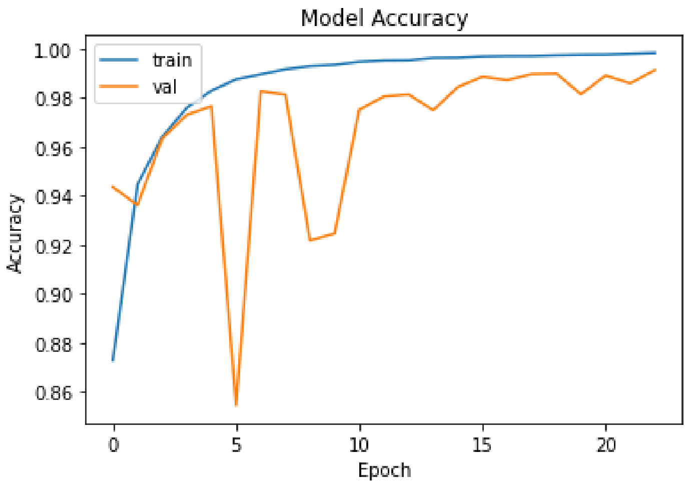

Figure 4. Performance curve of ResNet50

5.5 Case II Experiments

In Case II we extracted training and test samples from different datasets to measure the ability of the deepfake detection solutions to generalize.

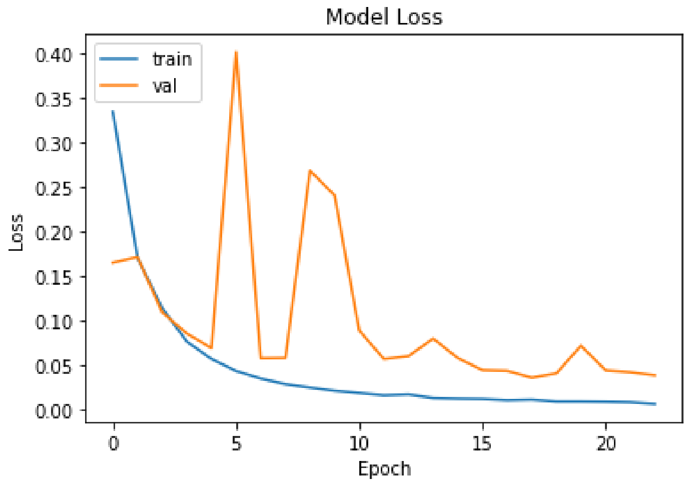

Figure 5. Optimization curve of ResNet50

5.5.1 Experimental Setup

We performed numerous experiments with changing size of triplet batch, dimensions of input samples, base network and loss function. Here we report the best results based on optimized parameters. The input size used is $128 \times 128 \times 3$ and the triplet batch size selected is 32. Higher batch sizes show a performance degradation. The loss function has a significant impact on the quality of results. We elaborate that in detail here:

Loss Functions. Three loss functions $Loss_1$ (Equation (13)), $Loss_2$ (Equation (14)), and $Loss_3$ (Equation (15)) have been used for our experiments.

$$Loss_1 = \max\left[\sum_{i=1}^{N}\{\{f_i^a - f_i^p\}^2 - \{f_i^a - f_i^n\}^2\}, 0\right] \tag{13}$$

where $i = 1, 2, 3, \ldots, N$, N is the number of dimension of features, a anchor, p positive, and n negative samples. f^a, f^p, f^n show the extracted features of anchor, positive, negative samples, respectively. Using $Loss_1$ model achieves 61.6 % accuracy. With $Loss_1$ in case of negative value of $\sum_{i=1}^{N}\{\{f_i^a - f_i^p\}^2 - \{f_i^a - f_i^n\}^2\}$ term, the model always returns zero failing to trigger any weight update or learning. Negative values occur when the distance between positive and anchor samples is large. By adding a margin term we reduce the problem of information loss.

Margin based loss function $Loss_2$ is described in following Equation (14):

$$Loss_2 = \max\left[\sum_{i=1}^{N}\{\{f_i^a - f_i^p\}^2 - \{f_i^a - f_i^n\}^2\} + margin, 0\right]. \tag{14}$$

Here, margin is a constant parameter and set as 0.4 in learning process of the model. $Loss_2$ yields 64.46 % accuracy which is a minor improvement to $Loss_1$. However, both $Loss_1$ and $Loss_2$ suffer from the problem of vanishing gradient. To control the vanishing gradient problem we introduce log loss function. Siamese triplet log loss with exponents provides the highest accuracies. The loss function is given as in Equation (15):

$$Loss_3 = \log\left[1 + \exp\left\{\sum_{i=1}^{N}\{|f_i^a - f_i^p| - |f_i^a - f_i^n|\}\right\}\right]. \tag{15}$$

We can see a jump of over 30 % in accuracy compared to margin-based loss while using log loss. $Loss_3$ is a non-linear function using logarithmic log(.) and exponent exp(.); which are non-linear and monotonically increasing functions that returns the increasing values in case of a negative term. This enhances the gradient preventing the vanishing gradient problem of previous two loss functions. With the exponential value in $Loss_3$, the model learns even in case of negative values. The model thus creates an embedding vector that minimizes the $d(a, p)$ and maximizes the $d(a, n)$.

5.5.2 Results

While pre-trained models (as in Case I) do provide over 90 % accuracy when trained on similar test and training samples, they seem to show a significant degradation in performance when the samples vary. This can be attributed to the fact that the pre-trained models tend to learn the artifacts that are left behind by the deepfake generation algorithm – sometimes termed as 'artificial fingerprints' – and thus perform poorly when the algorithm changes. For better visualization we have provided Figure 6 where the x-axis illustrates the model and the y-axis holds the respective detection accuracies of PGAN deepfakes. The highest detection rate 17.81 % of VGG19.

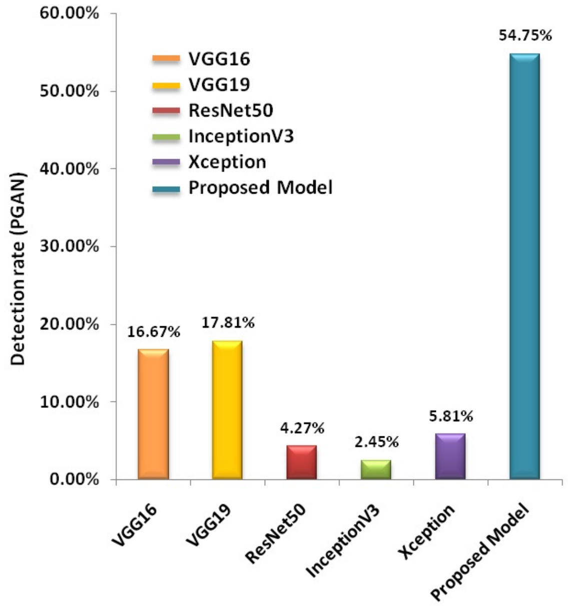

Figure 6. Performance of all methods evaluates with Case II

With such significant performance drop, we shifted our focus to developing an architecture that has the ability to better generalize by being less affected by the choice of deepfake algorithm. In this scenario a siamese network with triplet loss function creates optimal facial embeddings which show little to no connection with the generation algorithm. The results can be seen in Table 3. A visual representation of effect of siamese network on embedding vectors can be seen in figures.

Model	Case-I	Case-II (PGAN)
VGG16	94.70 %	16.67 %
VGG19	94.90 %	17.81 %
ResNet50	98.90 %	4.27 %
Inception V3	96.80 %	2.45 %
Xception	96.10 %	5.81 %
Proposed Method	94.80 %	54.75 %

Table 3. Comparison of accuracies of pre-trained networks and proposed model on Case II where training and test samples are from different datasets

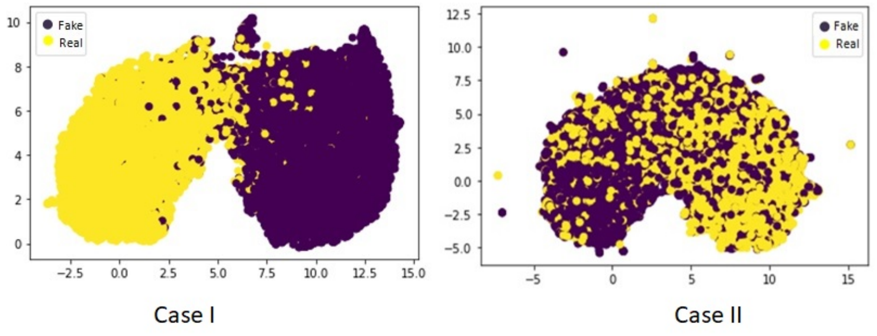

Figure 7. Siamese triplet loss embedding visualization

6 CONCLUSION

Deep learning generated images are often used for recreational purposes but their malicious uses have the ability to cause a serious disruption in the society. The need for a solution that has the ability to detect such deepfakes as-they-come is becoming more and more important by the day. Most existing solutions rely on the knowledge of the algorithms used to generate these images and thus show reduced performance in case of unseen examples.

In the proposed study we have used a siamese network in order to create representations of images that are independent on the generation algorithm. We provided extensive experiments with pretrained networks and our siamese network and have proved that the use of triplet loss at the time of feature extraction can make the features more robust. Triplet loss enforces the feature representations or embeddings have high similarity for related examples and vice-versa. We test multiple loss functions and different models to verify our findings. When test set is generated from a different dataset, our model provides a 54.75 % accuracy which is 3 times higher than its closest pre-trained model counterpart. This proves the claim that for better generalization networks need to be trained with not only regular loss functions as cross entropy or MSE but also with more suggestive loss functions as contrastive loss or triplet loss. This opens up a new dimension to how we have been training our neural networks traditionally.

REFERENCES

[1] Goodfellow, I. J.—Abadie, J. P.—Mirza, M.—Xu, B.—Farley, D. W.—Ozair, S.—Courville, A. C.—Bengio, Y.: Generative Adversarial Networks. 2014, doi: 10.48550/arXiv.1406.2661.

[2] Choi, Y.—Choi, M. J.—Kim, M.—Ha, J. W.—Kim, S.—Choo, J.: StarGAN: Unified Generative Adversarial Networks for Multi-Domain Image-to-Image Translation. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 8789–8797, doi: 10.1109/CVPR.2018.00916.

[3] Reed, S. E.—Akata, Z.—Yan, X.—Logeswaran, L.—Schiele, B.—Lee, H.: Generative Adversarial Text to Image Synthesis. In: Balcan, M. F., Weinberger, K. Q. (Eds.): Proceedings of the 33rd International Conference on Machine Learning. Proceedings of Machine Learning Research (PMLR), Vol. 48, 2016, pp. 1060–1069.

[4] Karras, T.—Laine, S.—Aila, T.: A Style-Based Generator Architecture for Generative Adversarial Networks. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 4396–4405, doi: 10.1109/CVPR.2019.00453.

[5] Kim, T.—Cha, M.—Kim, H.—Lee, J. K.—Kim, J.: Learning to Discover Cross-Domain Relations with Generative Adversarial Networks. In: Precup, D., Teh, Y. W. (Eds.): Proceedings of the 34th International Conference on Machine Learning. Proceedings of Machine Learning Research (PMLR), Vol. 70, 2017, pp. 1857–1865.

[6] Karras, T.—Aila, T.—Laine, S.—Lehtinen, J.: Progressive Growing of GANs for Improved Quality, Stability, and Variation. 6th International Conference on Learning Representations (ICLR 2018), Conference Track Proceedings, 2018, doi: 10.48550/arXiv.1710.10196.

[7] Yu, N.—Davis, L.—Fritz, M.: Attributing Fake Images to GANs: Learning and Analyzing GAN Fingerprints. 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 7555–7565, doi: 10.1109/ICCV.2019.00765.

[8] Güera, D.—Delp, E. J.: Deepfake Video Detection Using Recurrent Neural Networks. 2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), 2018, pp. 1–6, doi: 10.1109/AVSS.2018.8639163.

[9] Qi, P.—Cao, J.—Yang, T.—Guo, J.—Li, J.: Exploiting Multi-Domain Visual Information for Fake News Detection. 2019, doi: 10.48550/arXiv.1908.04472.

[10] Wang, R.—Ma, L.—Xu, F. J.—Xie, X.—Wang, J.—Liu, Y.: FakeSpotter: A Simple Baseline for Spotting AI-Synthesized Fake Faces. 2019, doi: 10.48550/arXiv.1909.06122.

[11] Afchar, D.—Nozick, V.—Yamagishi, J.—Echizen, I.: MesoNet: A Compact Facial Video Forgery Detection Network. 2018 IEEE International Workshop on Information Forensics and Security (WIFS), 2018, pp. 1–7, doi: 10.1109/WIFS.2018.8630761.

[12] Rössler, A.—Cozzolino, D.—Verdoliva, L.—Riess, C.—Thies, J.—Niessner, M.: FaceForensics++: Learning to Detect Manipulated Facial Images. 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 1–11, doi: 10.1109/ICCV.2019.00009.

[13] Tariq, S.—Lee, S.—Kim, H.—Shin, Y.—Woo, S. S.: Detecting Both Machine and Human Created Fake Face Images in the Wild. In: Hallman, R. A., Li, S., Chang, V. (Eds.): Proceedings of the 2nd International Workshop on Multimedia Privacy and Security (MPS '18). ACM, 2018, pp. 81–87, doi: 10.1145/3267357.3267367.

[14] Li, H.—Li, B.—Tan, S.—Huang, J.: Detection of Deep Network Generated Images Using Disparities in Color Components. 2018, doi: 10.48550/arXiv.1808.07276.

[15] Im, D. J.—Kim, C. D.—Jiang, H.—Memisevic, R.: Generating Images with Recurrent Adversarial Networks. 2016, doi: 10.48550/arXiv.1602.05110.

[16] Zhao, J. J.—Mathieu, M.—LeCun, Y.: Energy-Based Generative Adversarial Networks. 5th International Conference on Learning Representations (ICLR 2017), Conference Track Proceedings, 2017, doi: 10.48550/arXiv.1609.03126.

[17] van den Oord, A.—Kalchbrenner, N.—Kavukcuoglu, K.: Pixel Recurrent Neural Networks. In: Balcan, M. F., Weinberger, K. Q. (Eds.): Proceedings of the 33rd International Conference on Machine Learning. Proceedings of Machine Learning Research (PMLR), Vol. 48, 2016, pp. 1747–1756.

[18] Isola, P.—Zhu, J. Y.—Zhou, T.—Efros, A. A.: Image-to-Image Translation with Conditional Adversarial Networks. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 5967–5976, doi: 10.1109/CVPR.2017.632.

[19] Zhu, J. Y.—Park, T.—Isola, P.—Efros, A. A.: Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks. 2017 IEEE International Conference on Computer Vision (ICCV), 2017, pp. 2242–2251, doi: 10.1109/ICCV.2017.244.

[20] Wireless Lab: FaceApp. 2017, `https://www.faceapp.com` (Accessed: 2020-06-10).

[21] NEOCORTEXT, Inc.: Reflect. 2019, `https://reflect.tech` (Accessed: 2020-06-10).

[22] Mamo: ZAO. 2019, `http://www.zaoapp.com` (Accessed: 2020-06-10).

[23] Ledig, C.—Theis, L.—Huszár, F.—Caballero, J.—Cunningham, A.—Acosta, A.—Aitken, A.—Tejani, A.—Totz, J.—Wang, Z.—Shi, W.: Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 105–114, doi: 10.1109/CVPR.2017.19.

[24] Li, C.—Wand, M.: Precomputed Real-Time Texture Synthesis with Markovian Generative Adversarial Networks. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (Eds.): Computer Vision – ECCV 2016. Springer, Cham, Lecture Notes in Computer Science, Vol. 9907, 2016, pp. 702–716, doi: 10.1007/978-3-319-46487-9_43.

[25] Liu, M. Y.—Breuel, T.—Kautz, J.: Unsupervised Image-to-Image Translation Networks. In: Guyon, I., Von Luxburg, U., S., B., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (Eds.): Advances in Neural Information Process-

ing Systems 30 (NIPS 2017). Curran Associates, Inc., 2017, pp. 700–708, doi: 10.48550/arXiv.1703.00848.

[26] Li, Y.—Lyu, S.: Exposing DeepFake Videos by Detecting Face Warping Artifacts. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, 2019.

[27] Li, Y.—Chang, M. C.—Lyu, S.: In Ictu Oculi: Exposing AI Created Fake Videos by Detecting Eye Blinking. 2018 IEEE International Workshop on Information Forensics and Security (WIFS), 2018, pp. 1–7, doi: 10.1109/WIFS.2018.8630787.

[28] Yang, X.—Li, Y.—Lyu, S.: Exposing Deep Fakes Using Inconsistent Head Poses. ICASSP 2019 – 2019 IEEE International Conference on Acoustics, Speech and Signal Processing, 2019, pp. 8261–8265, doi: 10.1109/ICASSP.2019.8683164.

[29] Nguyen, T. T.—Nguyen, C. M.—Nguyen, D. T.—Nguyen, D. T.—Nahavandi, S.: Deep Learning for Deepfakes Creation and Detection. 2019, doi: 10.48550/arXiv.1909.11573.

[30] Cozzolino, D.—Thies, J.—Rössler, A.—Riess, C.—Niessner, M.—Verdoliva, L.: ForensicTransfer: Weakly-Supervised Domain Adaptation for Forgery Detection. 2018, doi: 10.48550/arXiv.1812.02510.

[31] Dang, H.—Liu, F.—Stehouwer, J.—Liu, X.—Jain, A.: On the Detection of Digital Face Manipulation. 2019, doi: 10.48550/arXiv.1910.01717.

[32] Do, N. T.—Na, I. S.—Kim, S. H.: Forensics Face Detection from GANs Using Convolutional Neural Network. 2018 ISITC International Securities Association for Institutional Trade Communication, 2018, pp. 376–379.

[33] Xuan, X.—Peng, B.—Wang, W.—Dong, J.: On the Generalization of GAN Image Forensics. In: Sun, Z., He, R., Feng, J., Shan, S., Guo, Z. (Eds.): Biometric Recognition (CCBR 2019). Springer, Cham, Lecture Notes in Computer Science, Vol. 11818, 2019, pp. 134–141, doi: 10.1007/978-3-030-31456-9_15.

[34] Tariq, S.—Lee, S.—Kim, H.—Shin, Y.—Woo, S. S.: GAN Is a Friend or Foe?: A Framework to Detect Various Fake Face Images. In: Hung, C. C., Papadopoulos, G. A. (Eds.): Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing (SAC 2019). ACM, 2019, pp. 1296–1303, doi: 10.1145/3297280.3297410.

[35] Marra, F.—Gragnaniello, D.—Cozzolino, D.—Verdoliva, L.: Detection of GAN-Generated Fake Images over Social Networks. 2018 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR), 2018, pp. 384–389, doi: 10.1109/MIPR.2018.00084.

[36] Zhang, X.—Karaman, S.—Chang, S. F.: Detecting and Simulating Artifacts in GAN Fake Images. 2019 IEEE International Workshop on Information Forensics and Security (WIFS), 2019, pp. 1–6, doi: 10.1109/WIFS47025.2019.9035107.

[37] Dang, L. M.—Min, K.—Lee, S.—Han, D.—Moon, H.: Tampered and Computer-Generated Face Images Identification Based on Deep Learning. Applied Sciences, Vol. 10, 2020, No. 2, Art. No. 505, doi: 10.3390/app10020505.

[38] Korshunov, P.—Marcel, S.: Vulnerability of Face Recognition to Deep Morphing. 2019, doi: 10.48550/arXiv.1910.01933.

[39] Gandhi, R.: Siamese Network and Triplet Loss. 2018, `https://towardsdatascience.com/siamese-network-triplet-loss-b4ca82c1aec8` (Ac-

cessed: 2020-08-07).

[40] Doshi, S.: Various Optimization Algorithms for Training Neural Network. 2019, `https://towardsdatascience.com/optimizers-for-training-neural-network-59450d71caf6` (Accessed: 2020-06-10).

[41] Szegedy, C.—Vanhoucke, V.—Ioffe, S.—Shlens, J.—Wojna, Z.: Rethinking the Inception Architecture for Computer Vision. 2015, doi: 10.48550/arXiv.1512.00567.

[42] Chollet, F.: Xception: Deep Learning with Depthwise Separable Convolutions. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 1800–1807, doi: 10.1109/CVPR.2017.195.

[43] He, K.—Zhang, X.—Ren, S.—Sun, J.: Deep Residual Learning for Image Recognition. 2015, doi: 10.48550/arXiv.1512.03385.

[44] False Positive Rate. `https://www.split.io/glossary/false-positive-rate` (Accessed: 2020-06-30).

Sidra Kanwal is an M.Sc. student at the Bahria University, Islamabad, Pakistan. Her area of research is digital image processing and AI.

Samabia Tehsin is currently working as Senior Associate Professor at the Bahria University, Pakistan. Her research interests include image analysis, machine learning, and multimedia forensics. She received her Ph.D. degree in computer software engineering, with specialization in digital image analysis, from the National University of Science and Technology, Islamabad, Pakistan.

Shahela Saif is researcher associated with the Bahria University, Islamabad, Pakistan. Her research interests include computer vision, machine learning, and image and video analysis. She received her M.Sc. degree in computer software engineering from the National University of Science and Technology, Islamabad, Pakistan.