On Hallucinating Context and Background Pixels from a Face Mask using Multi-scale GANs

Sandipan Banerjee*[1], Walter J. Scheirer[2], Kevin W. Bowyer[2], and Patrick J. Flynn[2]
[1] Affectiva, USA
[2] Department of Computer Science & Engineering, University of Notre Dame, USA
sandipan.banerjee@affectiva.com {wscheire, kwb, flynn}@nd.edu

Abstract

We propose a multi-scale GAN model to hallucinate realistic context (forehead, hair, neck, clothes) and background pixels automatically from a single input face mask, without any user supervision. Instead of swapping a face on to an existing picture, our model directly generates realistic context and background pixels based on the features of the provided face mask. Unlike facial inpainting algorithms, it can generate realistic hallucinations even for a large number of missing pixels. Our model is composed of a cascaded network of GAN blocks, each tasked with hallucination of missing pixels at a particular resolution while guiding the synthesis process of the next GAN block. The hallucinated full face image is made photo-realistic by using a combination of reconstruction, perceptual, adversarial and identity preserving losses at each block of the network. With a set of extensive experiments, we demonstrate the effectiveness of our model in hallucinating context and background pixels from face masks varying in facial pose, expression and lighting, collected from multiple datasets subject disjoint with our training data. We also compare our method with popular face inpainting and face swapping models in terms of visual quality, realism and identity preservation. Additionally, we analyze our cascaded pipeline and compare it with the progressive growing of GANs, and explore its usage as a data augmentation module for training CNNs.

1. Introduction

Generative adversarial nets (GANs) have revolutionized face synthesis research with algorithms being used to generate high quality synthetic face images [62, 9, 35, 36] or artificially edit visual attributes of existing face images like age [19, 2], pose [67, 75, 28], gender, expression and hairstyle [7, 54, 25]. However, these models require the full face image, comprising of the actual face, the context (forehead, hair, neck, clothes) and background pixels, to work. They fail to generate plausible results when the context and background pixels are absent (*i.e.*, when only the face mask is present). Facial inpainting models [44, 73, 29, 13, 72, 43, 55] that inpaint 'holes' work well when the missing pixels are small in number, located on or near the face. They do not generate realistic results when all of the context and background pixels are masked, as demonstrated in [69] and the experiments in Section 4 of this paper. As a potential solution, we propose a cascaded GAN model that requires only a few thousand training face images to generate realistic synthetic context and background pixels from face masks with different gender, ethnicity, lighting, pose and expression, across different datasets. Our model can be used to generate - (1) supplemental training data for CNNs, adding variety to the hair and background for real subjects or synthetic face masks generated by [49, 5] (section 4.4 of this paper), and (2) stock images for media usage without any copyright and privacy concerns.

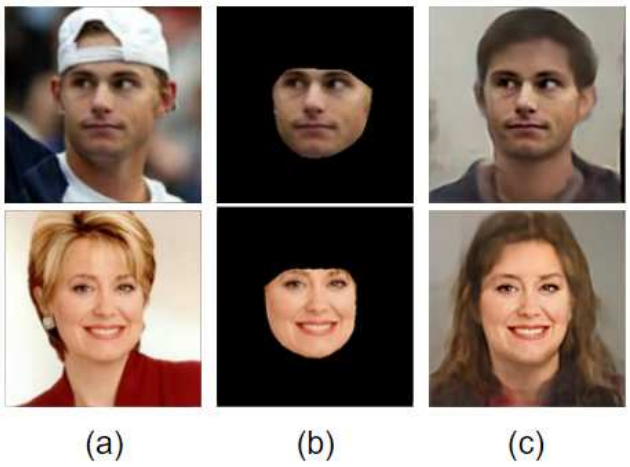

Figure 1: Our model, instead of swapping faces or inpainting missing facial pixels, directly hallucinates the entire context (forehead, hair, neck, clothes) and background from the input face mask. Sample results - (a) original face images from LFW [27] (2D aligned), (b) corresponding face masks (input), and (c) the hallucinated output generated by our cascaded network of GANs trained on [59]. All images are 128×128 in size.

* This work was done while SB was at Notre Dame

During training, our model takes as input a face image and its masked version, 128×128 in size, and downsamples both to their 64×64, 32×32, 16×16, and 8×8 versions. Training starts at the lowest GAN block (block_8), where it learns to reconstruct the 8×8 full face image from the corresponding 8×8 masked input. The output of this network is then upscaled 2x using a pixel shuffling block [65] and passed to the next GAN block (block_16). Thus instead of masked black pixels, block_16 receives a 16×16 input with roughly hallucinated context and background pixels, guiding it towards the direction of correct reconstruction. Its 16×16 output is then upscaled and sent to block_32 and so on (see Figure 2). At each block, we independently learn to hallucinate context and background pixels through reconstruction loss, adversarial loss provided by a discriminator, perceptual loss from [78] and an identity preserving loss using the pre-trained VGG-Face model [57]. During testing we only use the trained generator and pixel shuffling blocks to hallucinate the final 128×128 full face image from an input face mask. Sample results can be seen in Figure 1.

We perform the following experiments to assess the effectiveness of our model:

1. To gauge the effectiveness of our model in generating identity preserving, natural looking and diverse set of images we - (a) perform face matching experiments on [27] using the ResNet-50 model [24], (b) calculate SSIM [70] and perceptual error [61] values, and (c) the FID [26] between original and hallucinated images.

2. Using the above metrics, we compare our model with popular facial inpainting algorithms - **GenFace** [44], **DeepFillv1** [72], **SymmFCNet** [43], and **EdgeConnect** [55].

3. We compare our model with the popular **DeepFake**[1] face swapping application. Since it works only with tight face crops from a single identity, we train it on the LFW[27] subject, *George_W_Bush*, with the highest number of images (530). The trained network is used to synthesize source face crops, which are blended in the target face images.

4. We compare our single pass cascaded network with its progressively growing (ProGAN) version [35], where initial set of layers in the generator model are learned for a number of training epochs at the lowest resolution (8×8), and then we add new layers to learn hallucination at a higher resolution (16×16) and so on.

5. Using the CASIA-WebFace [74] dataset, we evaluate the potential usage of our model as a data augmentation module for training CNNs.

The main contributions of our paper are as follows:

1. We propose a method that can automatically synthesize context and background pixels from a face mask, using a cascaded network of GAN blocks, without requiring any user annotation. Each block learns to hallucinate the masked pixels at multiple resolutions (8×8 to 128×128) via a weighted sum of reconstruction, adversarial, identity preserving and perceptual losses. Trained with a few thousand images, it can hallucinate full face images from different datasets with a wide variety in gender, ethnicity, facial pose, expression and lighting.

2. We compare our model with recently proposed facial inpainting models [44, 72, 43, 55] and the DeepFake face swapping software. Our model generates photo-realistic results that produce higher quality scores (identity preservation, realism and visual quality) compared to the other algorithms on LFW [27].

3. We analyze the differences between the end-to-end training of our cascaded model with the ProGAN training regime from [35] while keeping the network architecture, and factors like training data, hyper parameters, and loss function fixed. We show the cascaded architecture to benefit the hallucination process and generate sharper results.

4. We evaluate the potential application of our model as a generator of supplemental training data for CNNs, to augment the intra-class variance by adding diverse hair and backgrounds to existing subjects of the dataset. When trained on this augmented data, we show the ResNet-50 model [24] to produce a boost in test performance.

2. Related Work

Face synthesis: While face synthesis research has greatly benefited from GANs [20, 62, 9, 35, 36], work in this domain began by simply combining neighborhood patches from different images to synthesize new faces [45, 3]. Other methods include expression and attribute flow for synthesizing new views of a face [52, 71]. Many works have also explored the use of a 3D head model to generate synthetic views of a face or frontalize it to an uniform setting [22, 49, 4, 5] while others have used GANs for this purpose [28, 67, 75]. Researchers have also used deep learning models to reconstruct face images from their rough estimates [16, 69, 6] or with new attributes altogether [7, 25, 14].

Face swapping: The first face swapping pipeline was proposed in [10], where a face is de-identified by blending together facial parts from other images. Many methods have modified this idea of recombining facial parts to generate synthetic images for de-identification or data augmentation [53, 3, 37]. In [56], a 3D morphable model based shape estimation is used to segment the source face and fit it to the target image prior to blending. Instead of using facial textures, the method in [54], uses latent variables from a deep network for face swapping. A style transfer [18] based face swapping approach was proposed in [40]; but it requires the network to be trained on only one source subject at a time. DeepFake is another recent method for face swapping, where an autoencoder is trained to reconstruct tight face crops of a subject from its warped versions. This

[1] https://en.wikipedia.org/wiki/Deepfake

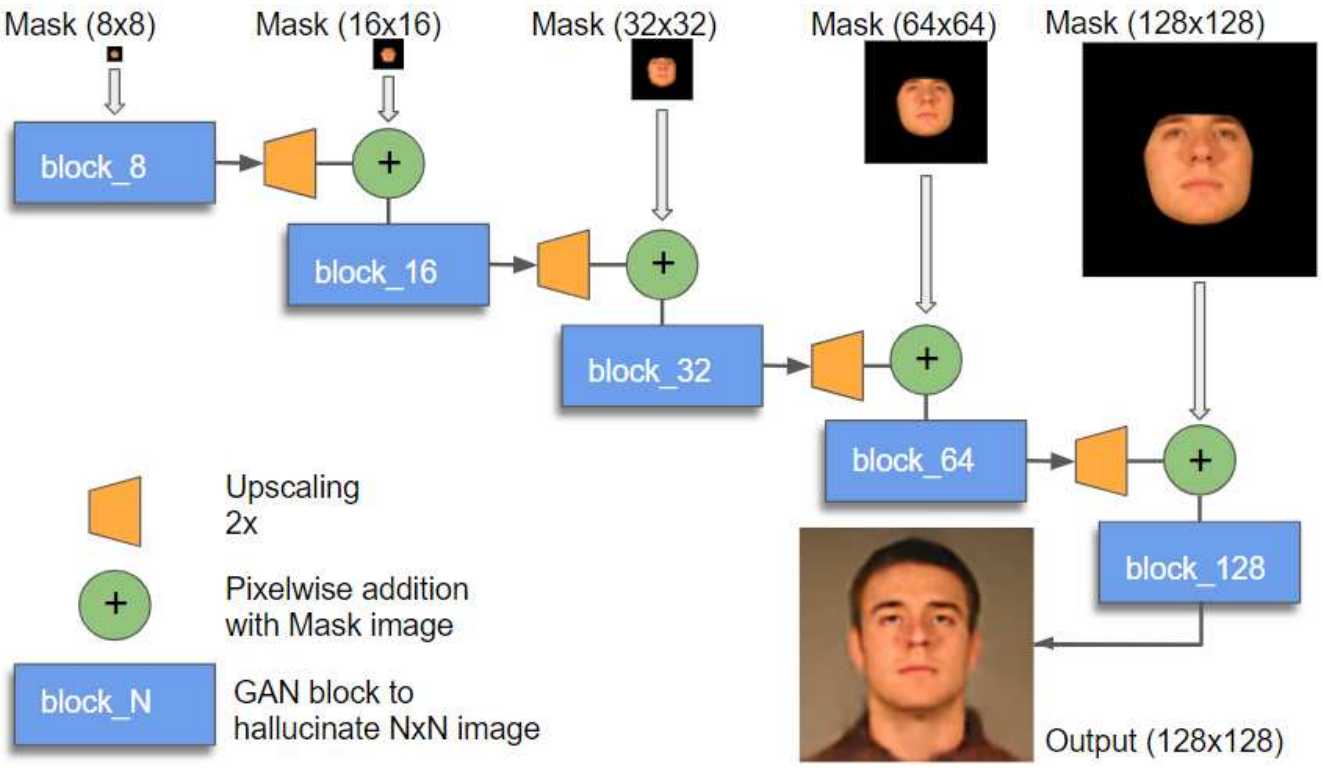

Figure 2: Our multi-scale cascaded network pipeline. Starting from the lowest resolution block (8×8), we proceed higher up through a set of GAN blocks in a single pass (left to right in the figure). Except the last block, the output of each block is upscaled 2x and fed as input to the next block. To preserve fine facial details at each resolution, we add the mask image at each resolution before feeding the input. The final 128×128 output, with hallucinated context and background pixels, is generated by block_128. More details about the architecture of block_128 is provided in Figure 3.

trained autoencoder is then used to hallucinate the source subject from different target face images. However, it works with one subject at a time and requires the target images to be highly constrained in visual attributes making it impractical for many real world applications.

Face inpainting: Image inpainting started with [8] transferring low-level features to small unknown regions from visible pixels. In [51, 31], this idea is used to reconstruct facial parts in missing regions using a positive, local linear representation. A simple inpainting scheme was proposed in [31], which uses features like ethnicity, pose and expression to fill missing facial regions. GANs have also been used for image completion, *e.g.* in [29, 44], a generator is used to hallucinate masked pixels, with discriminators and parser networks refining the results. In [73, 72, 55], information from the available data, surrounding image features, and edge structures are used for inpainting respectively. Facial symmetry is directly enforced in [43] to improve global consistency. In [60, 33], the inpainting process is guided by a rough sketch provided by the user. All these methods work well with small targeted masks[79], located on or near the face region, but perform poorly when a large masked area is presented[69], like the full context and background.

When supplied with a face mask (*i.e.*, limited data) the goal of our model is to automatically hallucinate realistic context and background pixels. While doing so the gender, ethnicity, pose, expression of the input subject should be preserved. While face swapping [40, 56, 54] and face editing [7, 25] algorithms have dealt with transferring the face and facial attributes from one identity to another, they require - (1) the full face image to work, and (2) similarity in visual appearance, and pose for identity preservation. Unlike previous work, we treat this problem along the same lines as image colorization [77, 42] and directly hallucinate the missing pixels taking cues from the input data without any involvement from the user.

3. Our Method

Since there can be many plausible hallucinations from a single face mask, we control this unconstrained problem using the training data. When provided with a face mask I^M during training, our model tunes its weights $\mathbf{w}$ such that its generated output $G(I^M)$ looks similar to the original face image I^{GT}. The weights are parameterized by I^{GT} itself and after a few training epochs, the model learns to generate $G(I^M)$ closely identical to I^{GT}. During testing, this trained model requires only a face mask (I^M), and not the full face image (I^{GT}), to hallucinate realistic context and background pixels from the learned representations.

3.1. Network Architecture

Cascaded Network. Inspired by [17, 68, 40], we implement a multi-scale architecture comprising of five GAN blocks to learn hallucination at multiple resolutions (8×8 to 128×128), as depicted in Figure 2. Unlike prior cascaded architectures, our model learns to hallucinate context and background for different image resolutions through a combination of multiple losses. Each block contains an encoder-decoder pair working as the generator. The encoder at the highest resolution block 'block_128', as shown in Figure 3, takes the input and downsamples it through a set of strided convolution layers (stride = 2), except the first layer where we encapsulate extra spatial information using an atrous convolution layer [76] with dilation rate of 2. Each of the next strided convolution layers is followed by a residual block [24] to facilitate the learning process. The output of the encoder is fed to the decoder which is composed of five convolution and pixel shuffling blocks [65] for upscaling the feature by two in each dimension.

We add skip connections [63, 24, 28] between encoder and decoder layers with the same tensor shape to propagate finer details from the input. The final 3 channel output is obtained by passing the upsampled result through a convolution layer with *tanh* activation [62, 64]. Since the input and output of 'block_(N/2)' is half in height and width compared to 'block_N', each GAN block contains one fewer residual and pixel shuffling layers than its next GAN block. Except 'block_128', the output of each block is upscaled 2x through a pixel shuffling layer and fed as input to the next block. Thus, instead of a face mask, the block receives a rough hallucination to guide it towards the right direction. For all blocks, we also replace pixels in the face mask region of $G(I^M)$ with original pixels from I^M, before loss computation, to keep finer details of the face intact and focus only on the task of context and background generation.

During training, we provide each block with a discriminator to guide the generated samples towards the distribution of the training data. We use the popular *CASIA-Net* architecture from [74] as the discriminator, after removing all max pooling and fully connected layers and adding batch normalization [30] to all convolution layers except the first one. A leaky *ReLU* [50] activation (slope = 0.2) is used for all layers except the last one where the *sigmoid* activation is adopted to extract a probability between 0 (fake) and 1 (real), as suggested by [62]. Each layer is initialized using He's initializer [23, 35]. During testing, only the trained generator and pixel shuffling blocks are used to hallucinate the synthetic output, with resolution of 128×128.

Progressively Growing Network (ProGAN). Addressing the recently proposed progressive growing of GANs to generate high quality samples [35, 13, 36], we also develop a ProGAN version of our model for comparison. Instead of the cascaded architecture where all the GAN blocks are trained in each iteration, we train the lowest resolution block_8 first with 8×8 face masks. After a few training epochs, we stop and load additional layers from block_16 and start training again with 16×16 face masks. This process of progressively growing the network by stopping and resuming training is continued till we have a trained block_128 model, as depicted in Figure 4. During testing, the trained block_128 is used to hallucinate context and background pixels directly from previously unseen 128×128 face masks. To maintain consistency, the loss function, hyper parameters and training data are kept the same with our cascaded network.

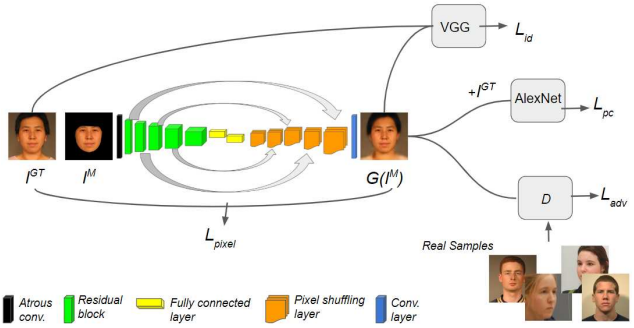

Figure 3: block_128 architecture. The encoder is composed of five residual blocks while the decoder upsamples the encoded feature using five pixel shuffling blocks. The solid curved arrows between layers represent skip connections. During training the generator learns to hallucinate the original full face image I^{GT} from the face mask I^M via reconstruction, identity preserving, perceptual and adversarial losses. We replace pixels in the face mask of $G(I^M)$ with original pixels from I^M to preserve fine details.

3.2. Loss Function

For each block of our network we learn context and background hallucinations independently. So we assign a combination of different losses, described below, to make the synthesized output at each resolution both realistic and identity preserving. We represent the image height, width and training batch size as H, W and N respectively.

1. **Pixel loss** (L_{pixel}): To enforce consistency between the pixels in the ground truth I^{GT} and hallucinated face images $G(I^M)$, we adopt a mean l_1 loss computed as:

$$L_{pixel} = \frac{1}{N \times H \times W} \sum_{n=1}^{N} \sum_{i=1}^{H} \sum_{j=1}^{W} \left| (I_n^{GT})_{ij} - (G(I_n^M))_{ij} \right| \tag{1}$$

where H and W increase as we move to higher blocks in our network, 8×8 → 16×16, 16×16 → 32×32, and so on. We use l_1 loss as it preserves high frequency signals better than l_2 in the normalized image thus generating sharper results.

2. **Perceptual loss** (L_{pc}): To make our hallucinations perceptually similar to real face images, we add the **LPIPS** metric (ver. 0.0) from [78] to our loss function. This metric finds a dissimilarity score between a pair of images, derived from deep features with varying levels of supervision, and

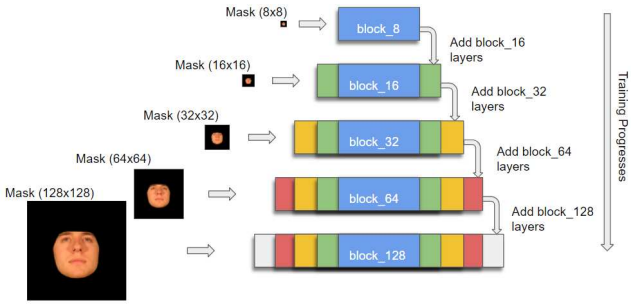

Figure 4: Pipeline of our progressively growing (ProGAN) network. We train the lowest resolution block for 50 epochs, then introduce additional layers for the next resolution block and resume training. This network growing continues till block_128. During testing, we only use the trained block_128.

is shown to be more consistent with human perception than classic similarity metrics like PSNR and SSIM [70]. We use LPIPS as a regularizer to support L_{pixel}. It is computed as:

$$L_{pc} = \frac{1}{N}\sum_{n=1}^{N} LPIPS(G(I_n^M), I_n^{GT}) \quad (2)$$

where $LPIPS$ is the dissimilarity score generated by the AlexNet [41] model[2] (in PyTorch [58]) provided by the authors. An L_{pc} value of 0 suggests perfect similarity between $G(I^M)$ and I^{GT}. Since the code does not support low-res images, L_{pc} is not applied on 'block_8' and 'block_16'.

3. **Adversarial loss** (L_{adv}): To push our hallucinations towards the manifold of real face images, we introduce an adversarial loss. This is achieved by training a discriminator along with the generator (encoder-decoder) at each block of our network. We use a mean square error based LSGAN [47] for this work as it has been shown to be more stable than binary cross entropy [20]. The loss is calculated as:

$$L_{adv} = \frac{1}{N}\sum_{n=1}^{N} \left(D(G(I_n^M)) - c\right)^2 \quad (3)$$

where D is the discriminator and c is set to 1 as we want to fool D into labeling the synthetic images as real.

4. **Identity loss** (L_{id}): To preserve essential features of the identity in the input face mask in the generated output, we use the pre-trained VGG-Face [57] model to provide a supporting metric. We calculate the l_2 distance between the *fc7* layer features between I^{GT} and $G(I^M)$ and apply that as content loss similar to neural style transfer [18]. The closer this metric moves towards 0, the better the hallucination quality. The loss is calculated as:

$$L_{id} = \frac{1}{N \times \#F}\sum_{n=1}^{N}\sum_{i=1}^{\#F}(F(G(I_n^M))_i - F(I_n^{GT})_i)^2 \quad (4)$$

[2]Available here: https://github.com/richzhang/PerceptualSimilarity

where F is the 4096-D feature vector from VGG-Face [57].

5. **Total variation loss** (L_{tv}): Similar to [34, 28, 40], we add a total variation loss as a regularizer to suppress spike artifacts, calculated as:

$$L_{tv} = \sum_{i=i}^{H}\sum_{j=1}^{W}(G(I^M)_{i,j+1} - G(I^M)_{i,j})^2 + (G(I^M)_{i+1,j} - G(I^M)_{i,j})^2 \quad (5)$$

The final loss L is computed as the weighted sum of the different losses:

$$L = L_{pixel} + \lambda_1 L_{pc} + \lambda_2 L_{adv} + \lambda_3 L_{id} + \lambda_4 L_{tv} \quad (6)$$

4. Experiments

Training Data. For training our model, we randomly sample 12,622 face images (7,761 male and 4,861 female) from the public dataset in [59]. These images were acquired specifically for recognition tasks, with variety of facial pose and neutral background. Image mirroring is then applied for data augmentation. To acquire the face masks, we first detect the face region using Dlib [38] and estimate its 68 facial keypoints with the pre-trained model from [11]. We remove images that Dlib fails to detect a face from. The eye centers are then used to align the faces and pixels outside the convex hull of the facial landmark points in the aligned image is masked. Both the aligned and masked versions are then resized using bilinear interpolation to 8×8×3, 16×16×3, 32×32×3, 64×64×3 and 128×128×3, with pixels normalized between [0,1], for training different network blocks.

Training Details. We train our model with the Adam optimizer [39] with generator and discriminator learning rates set as 10^{-4} and 2×10^{-4} respectively. For each block, we train its discriminator with separate real and synthesized mini-batches with label smoothing applied to the real mini-batch, as suggested by [62, 64]. Other hyper-parameters are set empirically as $\lambda_1 = 1$, $\lambda_2 = 0.1$, $\lambda_3 = 10$, $\lambda_4 = 10^{-6}$. We train our model on the NVIDIA Titan Xp GPU, using Tensorflow [1] and Keras [15], with a batch size of 10, for a hard limit of 50 epochs, as we find validation loss to plateau around this stage. We use the trained generator and pixel shuffling blocks from this model for our experiments.

Metrics for Quality Estimation. To evaluate the effectiveness of our model in the task of context and background hallucination, and compare with other works, we use the following metrics:

(1) **Mean Match Score**: We use the 256-dimensional penultimate layer descriptor from the 'ResNet-50-256D' model [24]('ResNet-50' here on), pre-trained on VGGFace2[12][3], as feature representation for an image for all our face recognition experiments. The deep features

[3]Available here: https://github.com/ox-vgg/vgg_face2

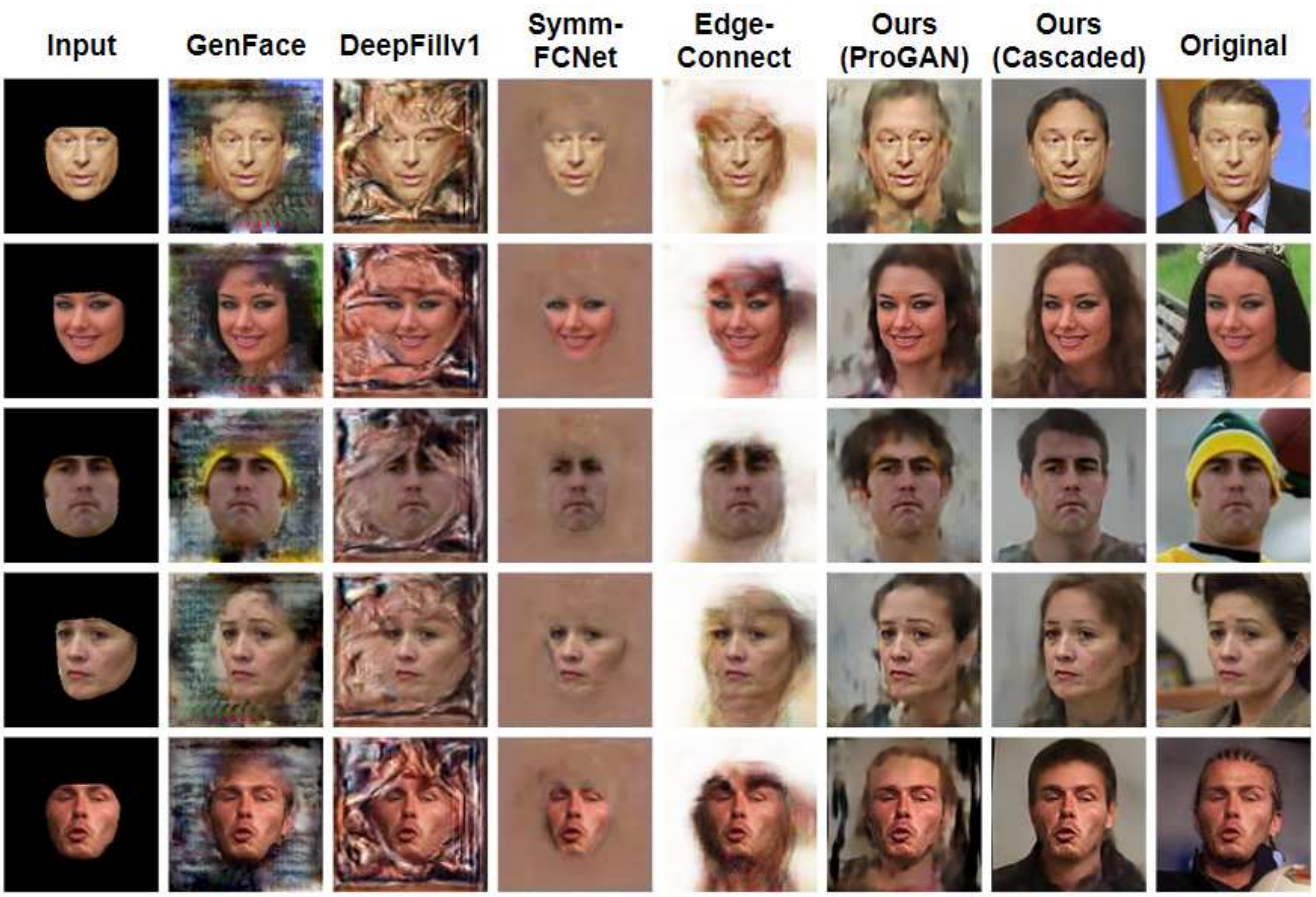

Figure 5: Sample results from LFW [27] (128×128 in size), generated using GenFace [44], DeepFillv1 [72], SymmFCNet [43], EdgeConnect [55], and our cascaded and ProGAN [35] models. Note the variation in gender, pose, age, expression and lighting in the input images.

are extracted for each original image and the hallucinated output in the dataset. The mean match score ρ is calculated by averaging the Pearson correlation coefficient between each feature pair as:

$$\rho = \frac{1}{N}\sum_{i=1}^{N}\frac{Cov((F_o)_i,(F_h)_i)}{\sigma_{(F_o)_i}\sigma_{(F_h)_i}} \tag{7}$$

where *Cov* denotes covariance, N is the number of images in the dataset, and $(F_o)_i$ and $(F_h)_i$ are the feature vectors of the i-th original and hallucinated images respectively. Ideally, we would like the hallucinated images to match well, but not perfectly, with the original images *i.e.*, ρ should be a little less than 1. Such a value would suggest that our model retains vital facial features of the input identity while adding variations in its visual attributes. The more the source face is modified, the more the gap widens, as specified in [56].

(2) **Mean SSIM**: To evaluate the degree of degradation, or noise, in the hallucinated output, we compute the SSIM [70] value for each (original,synthetic) image pair in the dataset. A higher mean SSIM value suggests less noisy hallucinations and therefore a better model.

(3) **FID**: To evaluate the realism of the generated samples, we use the Frechet Inception Distance (FID) metric proposed in [26]. FID uses activations from the Inception-v3 [66] network to compare the statistics of the generated dataset to the real one. A lower FID suggests generated samples to be more realistic, and signifies a better model.

(4) **Mean Perceptual Error**: To evaluate the perceptual dissimilarity between the original and the hallucinated images, we use the **PieAPP** v0.1 metric using the pre-trained model from [61]. The metric calculates the level of distortion between a pair of images, using a network trained on human ratings. A lower mean perceptual error indicates less noise in the hallucinated output, therefore a better model.

4.1. Comparison with Facial Inpainting Models

To gauge how our model compares with algorithms for generating missing pixels, we make use of four popular facial inpainting models: **GenFace** [44], **DeepFillv1** [72], **SymmFCNet** [43], and **EdgeConnect** [55]. We choose these models for our experiments, as - (1) they are open source with a pre-trained (on face images from CelebA [46]) models available for use, unlike [29, 13, 60], (2) they

Table 1: Quantitative results on the LFW [27] dataset.

Model	Mean Match Score	Mean SSIM [70]	FID [26]	Mean Perceptual Error [61]
GenFace [44]	0.543	0.491	177.06	3.536
DeepFillv1 [73]	0.481	0.321	241.696	3.204
SymmFCNet [43]	0.457	0.333	207.117	2.434
EdgeConnect [55]	0.454	0.178	141.695	3.106
DeepFake	0.459	0.448	**43.03**	1.857
Ours (ProGAN)	0.668	0.466	103.71	2.255
Ours (Cascaded)	**0.722**	**0.753**	46.12	**1.256**

can work with 128×128 face images, unlike [73], and (3) require no any user annotation, unlike [33].

To compare the models, we generate hallucinations using face masks from LFW [27]. Since each model is trained with different binary masks of missing pixels, we provide the model a binary mask with every pixel outside the face labeled as '0' instead of the actual masked face we feed to our trained model. Both qualitative and quantitative comparisons can be seen in Fig. 5 and Table 1 respectively. As shown in the table, our model (both versions) performs much better than the inpainting models for all metrics. These models aim to hallucinate the missing pixels, usually on or near the face region, using visual cues provided by facial pixels available in the image. Such cues are absent when whole of the context and background is masked, leading to noisy output. On the other hand, our model is specifically trained, and better suited for this task.

4.2. Comparison with DeepFake Face Swap

Owing to its huge popularity, we compare our model against the **DeepFake** face swapping application. The software essentially trains an autoencoder to learn transformations to change an input face crop (target) to another identity (source) while keeping target visual attributes intact. Since this autoencoder learns transformations for one subject at a time, we train it using 64×64 tight face crops of 'George_W_Bush', the LFW[27] identity with the most images (530). The autoencoder[4] is trained for 10K iterations using these 530 images, following which it can be used to hallucinate images of 'George_W_Bush' from face crops of other subjects and then blended onto the target images. The results of such a face swapping process can be seen in Figure 6 where we swap 'George_W_Bush' face images onto the context and background of 'Colin_Powell'. We choose 'Colin_Powell' as the mean hypercolumn [21] descriptor of his images, using *conv*-[1_2,2_2,3_3,4_3,5_3] features from VGG-Face [57], is proximal to that of 'George_W_Bush'.

Although DeepFake produces plausible results (lower FID [26] in Table 1), it requires both the source and target subjects to have fairly similar skin tone, pose and expression. Without such tight constraints, artifacts at the boundary of the blending mask are present as can be seen in the top row of Figure 6 due to the difference in skin tone and absence of eyeglasses in the source identity. Our model, on the other hand, has no such constraints as it learns to hallucinate the full set of context and background pixels from the provided face mask itself. Also, our model achieves a higher mean match score than DeepFake suggesting that it preserves more discriminative features of the source in the hallucinated images while adding variations in appearance.

Figure 6: Top row - synthetic images generated using **DeepFake** where the face mask (rectangle) is from 'George_W_Bush' but the context and background are from real face images of 'Colin_Powell' (from LFW [27]). Bottom row - synthesized context and background, using our trained cascaded model, for some images of the subject 'George_W_Bush'.

4.3. Comparison with our ProGAN Model

For the progressively growing (ProGAN [35]) version of our model, we set a training interval of 50 epochs after which we add new layers to the current block and resume training. Compared to the 96.53 hours required to train our cascaded network, our ProGAN model requires 66.24 hours to complete the full training at all scales, when trained on the same Titan Xp GPU system. The absence of multi-scale training, upscaling between blocks and depth concatenations during each iteration is the reason behind its lower training time. At the end of training, we feed 128×128 face masks to block_128 and get the hallucinated face images at the same resolution. We compare our cascaded and ProGAN models using masked face images from LFW [27]; the quantitative results are shown in Table 1 and few qualitative samples can be seen in Figure 5.

[4]We use the implementation from the most popular repo: https://github.com/deepfakes/faceswap

Table 2: Distribution and performance of training datasets with and without augmentation using our model.

Training Data	**CW [74] Images** (Identities)	**Hallucinated Images** (Identities)	**LFW [27] Performance** (TPR@FPR = 0.01)
Dataset 1	494,414 (10,575)	0	0.963
Dataset 2	494,414 (10,575)	494,414 (10,575)	0.971

Although the ProGAN model hallucinates slightly sharper results than the cascaded model due to the absence of upscaling between GAN blocks, it suffers from blurry artifacts, especially in the hair. This can be attributed to the fact that we only use block_128 of the ProGAN model to synthesize the output directly at of 128×128 like the trained generator from a single resolution GAN. Since the hallucination process in the cascaded network is guided at each resolution by the previous block, such artifacts are less frequent in its case. This might also be the reason of the difference in FID and perceptual error values between the two models in Table 1.

4.4. Effectiveness as Supplemental Training Data

To evaluate if our model can be used to augment existing face image datasets, we perform a recognition experiment using the CASIA-WebFace (CW) dataset [74]. CW contains 494,414 face images of 10,575 real identities collected from the web. We align, mask and resize all the face images from CW using the same pre-processing steps as our training data. These masked images are then fed to our trained cascaded model to hallucinate synthetic context and background pixels. Since the identity of the input face mask is preserved in our model (as shown by the Mean Match Score in Table 1), we label the hallucinated image as the same class as the original input from CW, similar to [49, 48, 5]. In this way, we generate 494,414 synthetic images, with hallucinated context and background, from 494,414 existing images of 10,575 real identities. We prepare two training sets from the images - 1) a dataset containing 494,414 real images from CW and no synthetic images (Dataset 1 from Table 2), and 2) a dataset containing 494,414 real images and 494,414 synthetic images of the same 10,575 subjects (Dataset 2 from Table 2).

We fine-tune the ResNet-50 [24] model with these datasets in two separate training sessions, where 90% of the data is used for training and the rest for validation. The networks are trained using the Caffe [32] framework, with a base learning rate = 0.001 and a polynomial decay policy where gamma = 0.96, momentum = 0.009, and step size = 32K training iterations. We set the batch size = 16, and train each network till its validation loss plateaus across an epoch. After training terminates, we save its snapshot for testing on the LFW dataset [27]. Each image is passed to the snapshot and its 256-D vector is extracted from the penultimate (*feat_extract*) layer. We use these features to perform a verification experiment (all vs. all matching) with Pearson correlation for scoring, the results of which are presented in Table 2. As shown, the supplemental synthetic images introduce more intra-subject variation in context and background, which in turn slightly boosts the performance of the network during testing. Our trained model can therefore be used to augment existing face image datasets for training CNNs, especially to generate the diverse context and background pixels in synthetic face masks generated by [49, 5].

An extended version of this paper containing more qualitative results, architecture details, and impact of individual losses can be found here: https://arxiv.org/abs/1811.07104.

5. Conclusion

In this paper, we propose a cascaded network of GAN blocks that can synthesize realistic context and background pixels given a masked face input, without requiring any user supervision. Instead of swapping a source face onto a target image or inpainting small number of missing facial pixels, our model directly hallucinates the entire set of context and background pixels, by learning their representation directly from the training data. Each GAN block learns to hallucinate the missing pixels at a particular resolution via a combination of different losses and guides the synthesis process of the next block.

While trained on only 12K face images acquired at a controlled setting, our model is effective in generating on challenging images from the LFW [27] dataset. When compared with popular facial inpainting models [44, 72, 43, 55] and face swapping methods (DeepFake), our model generates more identity-preserving (evaluated using deep features from ResNet-50 [24]) and realistic (evaluated using SSIM [70], FID [26], and perceptual error [61]) hallucinations. Our model can also be used to augment training data for CNNs by generating different hair and background of real subjects [74] or rendered synthetic face masks using [49, 5]. This can increase the intra-class variation in the training set, which in turn can make the CNN more robust to changes in hair and background along with variations in facial pose and shape. The generated face images can also be used as stock images by the media without any privacy concerns.

A possible extension of this work would be to increase the resolution of the synthetic face images, possibly by adding more generator blocks to the cascaded network in a progressive manner [35, 13]. The soft facial features of the generated output can also be varied by adding style based noise to the generator [36], while keeping the subject identity constant. Implementing this scheme to work on full face videos could be another avenue to explore.

References

[1] M. Abadi and et al. Tensorflow: A system for large-scale machine learning. In *OSDI*, 2016. 5

[2] G. Antipov, M. Baccouche, and J. L. Dugelay. Face aging with conditional generative adversarial networks. *ICIP*, 2017. 1

[3] S. Banerjee, J. Bernhard, W. Scheirer, K. Bowyer, and P. Flynn. Srefi: Synthesis of realistic example face images. In *IJCB*, 2017. 2

[4] S. Banerjee, J. Brogan, J. Krizaj, A. Bharati, B. RichardWebster, V. Struc, P. Flynn, and W. Scheirer. To frontalize or not to frontalize? do we really need elaborate pre-processing to improve face recognition? *WACV*, 2018. 2

[5] S. Banerjee, W. Scheirer, K. Bowyer, and P. Flynn. Fast face image synthesis with minimal training. In *WACV*, 2019. 1, 2, 8

[6] A. Bansal, Y. Sheikh, and D. Ramanan. Pixelnn: Example-based image synthesis. *ICLR*, 2018. 2

[7] J. Bao, D. Chen, F. Wen, H. Li, and G. Hua. Towards open-set identity preserving face synthesis. In *CVPR*, 2018. 1, 2, 3

[8] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester. Image inpainting. In *SIGGRAPH*, 2000. 3

[9] D. Berthelot, T. Schumm, and L. Metz. Began: Boundary equilibrium generative adversarial networks. *arXiv:1703.10717*. 1, 2

[10] D. Bitouk, N. Kumar, S. Dhillon, S. Belhumeur, and S. K. Nayar. Face swapping: Automatically replacing faces in photographs. *SIGGRAPH*, 2005. 2

[11] A. Bulat and G. Tzimiropoulos. How far are we from solving the 2d & 3d face alignment problem? (and a dataset of 230,000 3d facial landmarks). In *ICCV*, 2017. 5

[12] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and A. Zisserman. Vggface2: A dataset for recognizing faces across pose and age. In *arXiv:1710.08092*. 5

[13] Z. Chen, S. Nie, T. Wu, and C. Healey. High resolution face completion with multiple controllable attributes via fully end-to-end progressive generative adversarial networks. *arXiv:1801.07632*, 2018. 1, 4, 6, 8

[14] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Chool. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *CVPR*, 2018. 2

[15] F. Chollet et al. Keras. https://github.com/fchollet/keras, 2015. 5

[16] F. Cole, D. Belanger, D. Krishnan, A. Sarna, I. Mosseri, and W. T. Freeman. Face synthesis from facial identity features. In *CVPR*, 2017. 2

[17] E. Denton, S. Chintala, A. Szlam, and R. Fergus. Deep generative image models using a laplacian pyramid of adversarial networks. In *NIPS*, 2015. 4

[18] L. Gatys, A. Ecker, and M. Bethge. A neural algorithm of artistic style. *arXiv:1508.06576*, 2015. 2, 5

[19] J. Gauthier. Conditional generative adversarial networks for convolutional face generation. In *Tech Report*, 2015. 1

[20] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio. Generative adversatial nets. In *NIPS*, 2014. 2, 5

[21] B. Hariharan, P. Arbelaez, R. Girshick, and J. Malik. Hypercolumns for object segmentation and fine-grained localization. In *CVPR*, 2015. 7

[22] T. Hassner, S. Harel, E. Paz, and R. Enbar. Effective face frontalization in unconstrained images. In *CVPR*, 2015. 2

[23] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV*, 2015. 4

[24] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CVPR*, 2016. 2, 4, 5, 8

[25] Z. He, W. Zuo, M. Kan, S. Shan, and X. Chen. Attgan: Facial attribute editing by only changing what you want. In *arXiv:1711.10678*, 2017. 1, 2, 3

[26] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *NeurIPS*, 2017. 2, 6, 7, 8

[27] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. In *Tech Report 07–49*, 2007. 1, 2, 6, 7, 8

[28] R. Huang, S. Zhang, T. Li, and R. He. Beyond face rotation: Global and local perception gan for photorealistic and identity preserving frontal view synthesis. *ICCV*, 2017. 1, 2, 4, 5

[29] S. Iizuka, E. Simo-Serra, and H. Ishikawa. Globally and locally consistent image completion. In *SIGGRAPH*, 2017. 1, 3, 6

[30] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015. 4

[31] M. Jampour, C. Li, L.-F. Yu, K. Zhou, S. Lin, and H. Bischof. Face inpainting based on high-level facial attributes. *CVIU*, 161:29–41, 2017. 3

[32] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *ACM MM*, 2014. 8

[33] Y. Jo and J. Park. Sc-fegan: Face editing generative adversarial network with user's sketch and color. In *arXiv:1902.06838*, 2019. 3, 7

[34] J. Johnson, A. Alahi, and F.-F. Li. Perceptual losses for real-time style transfer and super-resolution. In *ECCV*, 2016. 5

[35] T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *ICLR*, 2018. 1, 2, 4, 6, 7, 8

[36] T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019. 1, 2, 4, 8

[37] I. Kemelmacher-Shlizerman. Transfiguring portraits. *SIGGRAPH*, 2016. 2

[38] D. E. King. Dlib-ml: A machine learning toolkit. In *Journal of Machine Learning Research*, volume 10, pages 1755–1758, 2009. 5

[39] D. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 5

[40] I. Korshunova, W. Shi, J. Dambre, and L. Theis. Fast face-swap using convolutional neural networks. In *ICCV*, 2017. 2, 3, 4, 5

[41] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 5

[42] G. Larsson, M. Maire, and G. Shakhnarovich. Learning representations for automatic colorization. In *ECCV*, 2016. 3

[43] X. Li, M. Liu, J. Zhu, W. Zuo, M. Wang, G. Hu, and L. Zhang. Learning symmetry consistent deep cnns for face completion. In *arXiv:1812.07741*, 2018. 1, 2, 3, 6, 7, 8

[44] Y. Li, S. Liu, J. Yang, and M.-H. Yang. Generative face completion. In *CVPR*, 2017. 1, 2, 3, 6, 7, 8

[45] W. Liu, D. Lin, and X. Tang. Neighbor combination and transformation for hallucinating faces. In *ICME*, 2005. 2

[46] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *ICCV*, 2015. 6

[47] X. Mao, Q. Li, H. Xie, R. Lau, Z. Wang, and S. Smolley. Least squares generative adversarial networks. In *ICCV*, 2017. 5

[48] I. Masi, T. Hassner, A. T. Tran, and G. Medioni. Rapid synthesis of massive face sets for improved face recognition. *FG*, 2017. 8

[49] I. Masi, A. T. Tran, J. T. Leksut, T. Hassner, and G. Medioni. Do we really need to collect millions of faces for effective face recognition? In *ECCV*, 2016. 1, 2, 8

[50] A. Mass, A. Hannun, and A. Ng. Rectifier nonlinearities improve neural network acoustic models. In *ICML*, 2013. 4

[51] Z. Mo, J. Lewis, and U. Neumann. Face inpainting with local linear representations. In *BMVC*, 2004. 3

[52] U. Mohammed, S. J. D. Prince, and J. Kautz. Visio-lization: Generating novel facial images. *SIGGRAPH*, 2009. 2

[53] S. Mosaddegh, L. Simon, and F. Jurie. Photorealistic face de-identification by aggregating donors' face components. In *ACCV*, 2014. 2

[54] R. Natsume, T. Yatagawa, and S. Morishima. Fsnet: An identity-aware generative model for image-based face swapping. In *ACCV*, 2018. 1, 2, 3

[55] K. Nazeri, E. Ng, T. Joseph, F. Qureshi, and M. Ebrahimi. Edgeconnect: Generative image inpainting with adversarial edge learning. In *arXiv:1901.00212*, 2019. 1, 2, 3, 6, 7, 8

[56] Y. Nirkin, I. Masi, A. T. Tran, T. Hassner, and G. Medioni. On face segmentation, face swapping, and face perception. *FG*, 2018. 2, 3, 6

[57] O. M. Parkhi, A. Vedaldi, and A. Zisserman. Deep face recognition. In *BMVC*, 2015. 2, 5, 7

[58] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. 2017. 5

[59] P. J. Phillips, P. Flynn, and K. Bowyer. Lessons from collecting a million biometric samples. *Image and Vision Computing*, 2016. 1, 5

[60] T. Portenier, Q. Hu, A. Szabo, S. Bigdeli, P. Favaro, and M. Zwicker. Faceshop: Deep sketch-based face image editing. In *SIGGRAPH*, 2018. 3, 6

[61] E. Prashnani, H. Cai, Y. Mostofi, and P. Sen. Pieapp: Perceptual image-error assessment through pairwise preference. In *CVPR*, 2018. 2, 6, 7, 8

[62] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *ICLR*, 2016. 1, 2, 4, 5

[63] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015. 4

[64] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. In *NIPS*, 2016. 4, 5

[65] W. Shi, J. Caballero, F. Huszar, J. Totz, A. Aitken, R. Bishop, D. Rueckert, and Z. Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *CVPR*, 2016. 2, 4

[66] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, 2016. 6

[67] L. Tran, X. Yin, and X. Liu. Disentangled representation learning gan for pose-invariant face recognition. In *CVPR*, 2017. 1, 2

[68] D. Ulyanov, V. Lebedov, A. Vedaldi, and V. Lempitsky. Texture networks: Feed-forward synthesis of textures and stylized images. In *ICML*, 2016. 4

[69] P. Upchurch, J. Gardner, G. Pleiss, R. Pless, N. Snavely, K. Bala, and K. Weinberger. Deep feature interpolation for image content changes. *CVPR*, 2017. 1, 2, 3

[70] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. on Image Processing*, 13(4):600–612, 2004. 2, 5, 6, 7, 8

[71] F. Yang, J. Wang, E. Shechtman, L. Bourdev, and D. Metaxas. Expression flow for 3d-aware face component transfer. *SIGGRAPH*, 2011. 2

[72] J. Yang, X. Shen, X. Lu, and T. Huang. Generative image inpainting with contextual attention. In *CVPR*, 2018. 1, 2, 3, 6, 8

[73] R. Yeh, C. Chen, T. Lim, A. Schwing, M. Hasegawa-Johnson, and M. Do. Semantic image inpainting with deep generative models. In *CVPR*, 2017. 1, 3, 7

[74] D. Yi, Z. Lei, S. Liao, and S. Z. Li. Learning face representation from scratch. In *arXiv:1411.7923*. 2, 4, 8

[75] X. Yin, X. Yu, K. Sohn, X. Liu, and M. Chandraker. Towards large-pose face frontalization in the wild. *ICCV*, 2017. 1, 2

[76] F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. In *ICLR*, 2016. 4

[77] R. Zhang, P. Isola, and A. Efros. Colorful image colorization. In *ECCV*, 2016. 3

[78] R. Zhang, P. Isola, A. Efros, E. Shechtman, and O. Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 2, 4

[79] Y. Zhao, W. Chen, J. Xing, X. Li, Z. Bessinger, F. Liu, W. Zuo, and R. Yang. Identity preserving face completion for large ocular region occlusion. In *BMVC*, 2018. 3