Mask Removal Algorithm using GAN based Models

Liqin Ye
Information and Computer Science
University of California, Irvine
Irvine, United States
liqiny@uci.edu

Abstract—**Recent deep learning network models have been enhanced dramatically to generating images, removing objects, and improving resolution of images. From a pragmatic purpose, we must utilize it to solve some occurrent real-world problems, for example, facial mask removal. People's frequency of wearing masks has increased rapidly due to the pandemic. The objective of this research is to remove the facial mask on face images using generative adversarial network, or GAN, model. We set up three different training process to explore how to remove a facial mask effectively. By comparing among these training processes, we find out that adding an additional binary mask segmentation is conducive to both the generation quality of face region other than mask and inpainting quality of missing mask region. Moreover, our customized GAN model outperforms the Pix2Pix GAN model in both region inpainting.**

Keywords-GAN, Pix2Pix, facial mask removal, computer vision.

I. INTRODUCTION

This research aims to remove the facial mask on people's faces. Technically, it manages to achieve image inpainting on the specific mask region. COVID-19 brings everyone to live in a mask life. People's tendency to wear a mask in public places has reached a peak in recent years. Though wearing a mask gives us protection from various viruses, it is also really inconvenient for our daily life. With a mask, people are unable to pass the facial ID on their phone. Some facial recognition machines are not capable of verifying a masked face. Inspired by these dilemmas, this project aims to generate unmasked images of human faces based on their face images with facial masks using Generative Adversarial Network (GAN). To achieve this objective, we need to have a large size mask face dataset for training. Our dataset is a synthetic one in which we run a mask-putting algorithm to mask the facial images from a famous face image dataset. Along with the mask face, we also output a binary mask segmentation which indicates the region and position of the mask on the corresponding face. The deep learning model that we choose to solve this problem is a GAN-base model. Two different kinds of GAN models are implemented in our experiment. One is the Pix2Pix GAN model and the other is our customized GAN. We train them on different inputs and compare their consequences. Our experiment results show that adding an additional binary mask channel to the input facial image improves the generative quality of both mask region and rest face region. Also, our customized GAN model outperforms the Pix2Pix model in this facial mask removal tasks.

II. METHOD

A. General Model Introduction (GAN)

Among numerous deep learning methods, we choose the generative adversarial network[1], or GAN, to solve this problem. The generative adversarial network is actually a type of generative model, which is an approach of unsupervised learning. In a typical generative model, the training samples follow a specific distribution $P_{\text{data}}(x)$, and the generative model algorithm tries to learn and create a distribution $P_{\text{model}}(x)$ that is as similar to $P_{\text{data}}(x)$ as possible. Different from other classic generative models, the generative adversarial network has an additional structure, a discriminator which differentiates the generated $P_{\text{model}}(x)$ from $P_{\text{data}}(x)$. During training, both networks manage to minimize their losses. Equation1 shows the objective of two networks during the training process. The z in the equation is the input random noise for the generator, which will be modified to approach real data x. The training philosophy for discriminator is to maximize the probability of classifying real and fake data. As for the generator, the goal is to minimize log(1-$D(G(z))$), the probability that the generated images fool the discriminator. In general, GAN is embedded with two different networks competing against each other to produce the data. The additional discriminator is conducive for GAN to generate more realistic data. GAN's working mechanism and architecture makes it effective and powerful in image inpainting since every generated image will be compared to the real one in order to make potential improvements. With the help of it, we can restore the missing face under the mask more realistically.

$$\min_G \max_D V(G,D) = E_{x\sim p_{data}(x)}[\log D(x)] + E_{z\sim p_z(z)}\left[1-\log\left(D\left(G(z)\right)\right)\right] \qquad (1)$$

B. Dataset Generation

Our dataset is derived from Labeled Face in the Wild from the University of Massachusetts, which contains over 13,000 real face images. We use this publicly available resource to create a paired synthetic dataset. We use a mask-face algorithm to put different masks (21 templates in total shown in Figure 2), ranging from surgical mask to KN95 mask, on our training samples. With each image, the algorithm first detected landmarks, including positions of eyebrow, eyes, nose, mouth, and face contour. Based on this detected information, the algorithm will estimate the approximate mask positions on each face. The key position enables the algorithm to estimate the face' tilt angle and select the most appropriate mask template. Finally, the algorithm will wrap the mask and lay it over the face. Therefore, each of our training data is a paired image, original face and mask face. Moreover, as shown in Figure 1, we also output the corresponding binary mask segmentation of each mask face during the generation. The binary mask segmentation indicates the mask position and region in a specific mask face image. We will need this segmentation in later training process.

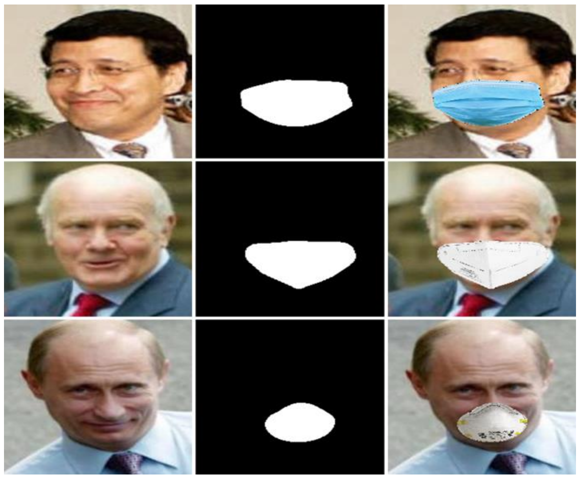
Figure 1: Examples of images in our training dataset

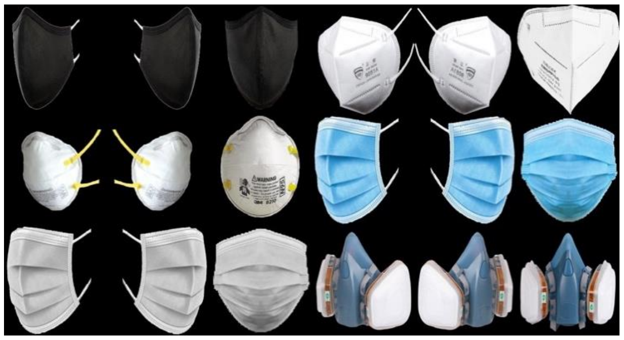
Figure 2: Examples of images in our training dataset

C. *Pix2Pix GAN Model*

There are multiple GAN models available for us to choose from. Our solution to the face-mask removal problem is the Pix2Pix generative adversarial network (GAN). Pix2Pix GAN, which is well-designed for various types of image-to-image translation tasks, is a type of conditional GAN, where the generated output image is conditionally based on the input. In our scenario, although the only region that we need to generate is the missing part under the mask, we have to make the generated mask region fit to the overall face. That's why we need a conditional GAN to achieve our goal of image inpainting. With a conditional input, the objective is slightly different from a regular GAN model. As we illustrate above, the generator in the regular GAN is to learn the mapping from a random vector z to the probability distribution of real data x, $P_{\text{data}}(x)$. In a conditional GAN, the generator is mapping z to the conditional probability distribution, $P_{\text{data}}(x|c)$. Similar to the generator, the discriminator in conditional GAN also takes in the conditional input c. The complete objective function of the conditional GAN is shown in the Equation 2 below.

$$\min_G \max_D V(G,D) = E_{c\sim p_{data}(c)}[E_{x\sim p_{data}(x|c)}[\log D(x)] + E_{z\sim p_z(z)}[1 - log(D(G(z,c),c))]] \quad (2)$$

Different from a regular conditional GAN, the architecture of the generator in Pix2Pix GAN is U-Net (Figure 3) [4], an encoder-decoder with skip connections between mirrored layers. The special skip connection between each corresponding layer in the generator offers a high-level information share between the output and the input, which ensures the consistency of several prominent characteristics in the input image. Our Pix2Pix GAN generator contains 8 encoders and 7 decoders, where 6 pairs of layers mirrored from the 8th encoders are connected with each other. As shown in Figure 4, all layers are convolutional layers with Leaky ReLU. Besides the first and eighth layers, all the other layers have batch normalization within. Moreover, the first three layers in the decoders are built with dropout (0.5).

Speaking of the discriminator, we use PatchGAN [4] to distinguish whether the image is real or fake. PatchGAN is a GAN discriminator which distinguishes the overall image at the scale of patches. It manages to distinguish if each N x N patch in the image is real or fake. The PatchGAN will be run convolutionally to the input image, where each N x N patch will output a result. The ultimate result of the image is the average of numerous N x N responses. Our discriminator is implemented with 4 convolutional layers with batch normalization and leaky ReLU (Figure 4).

The loss for generator (Equation 3) consists of adversarial loss and reconstruction loss, where adversarial loss is gained by applying Binary Cross-Entropy with Logits Loss (BCEwithLogitsLoss) [5] to the result of inputting generated image and real image to the PatchGAN. The reconstruction loss is the L1 loss of predicted image and real image. We assign a weight (200) to the reconstruction loss. As for the loss of discriminator (Equation 4), we follow the optimization method in this paper[2] by dividing the objective by 2. The optimizer we use is SGD and Adam Solver with learning rate 0.0002.

$$G_{loss} = BCEWithLogitsLoss(D(G(z,c),c)) + 200 \times L1(G(z,x),x) \quad (3)$$

$$D_{loss} = \frac{BCEWithLogitsLoss(D(G(z,c),c)) + BCEWithLogitsLoss(D(x,c))}{2} \quad (4)$$

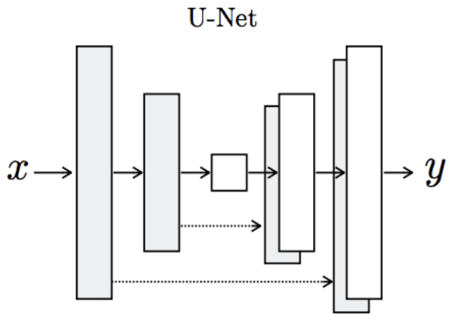

Figure 3: Basic U-Net Model Architecture

D. *Customized GAN Model*

We have also implemented another customized GAN model to our problem, where we modified the generator based on this research [6]. The overall structure of this customized GAN model is shown in Figure 5. We remain the U-Net architecture, but we reduced the number of connected layers from 7 to 5. Additionally, we add 4 dilation convolution layers [7] with dilation rate 2, 4, 8, 16 in sequence. The dilation convolution can enlarge the view of the kernel, making the generated missing mask region coherent with the rest of the face. Based on this paper [6], we add a squeeze and excitation block [8] after the first three encoders accordingly, which can improve the performance of convolutional layers. Instead of using batch normalization in each convolution layer, we use instant normalization. Also, we replace the L1 loss in the optimization of the generator with SSIM [9] loss, which penalizes the generated image by its dissimilarity with the real image. We remain all the other structure and procedure the same as our Pix2Pix GAN model. The new generator loss is shown in the equation 5 below.

$$G_{loss} = BCEWithLogitsLoss(D(G(z,c),c)) + 200 \times SSIM(G(z,x),x) \tag{5}$$

E. *Training Detail*

We have experimented with 3 training processes (Train A,B,C shown in Figure4 and Figure 5) by implementing different GAN models as well as different inputs. Our experiment can be regarded as 2 parts. In the first part, we adopt the Pix2Pix GAN model that we describe above. As for train A, we only input the 256 x 256 3-channel face image in our synthetic dataset. During train B, the input is changed to a 256x256 4-channel face image. The additional channel we add is the binary mask segmentation that we generate along with masked faces during the generation of our dataset. We concatenate the corresponding binary mask map with the face image and pass it through our GAN model. Our last experiment, train C, utilizes the customized GAN model. Similarly, the customized GAN model takes in a 4-channel face image with binary mask segmentation in. Besides the differences, we ran 50k iterations through each training process. The batch size we used for the Pix2Pix GAN model is 10, while the customized GAN model's is 4. The GPU we use is the NVIDIA GeForce GTX 1060. Our total training time for all three models is 32 hours.

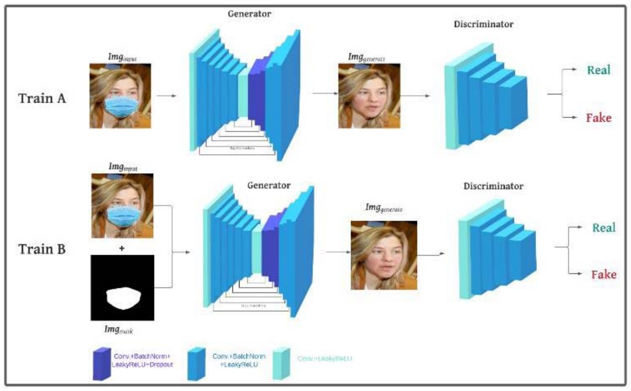

Figure 4: Pix2Pix GAN Training Process

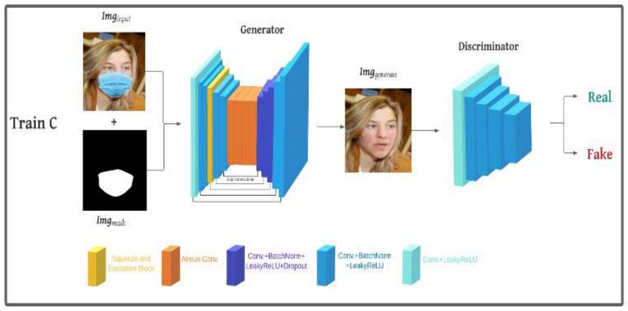

Figure 5: Customized GAN Training Process

III. RESULT AND DISCUSSION

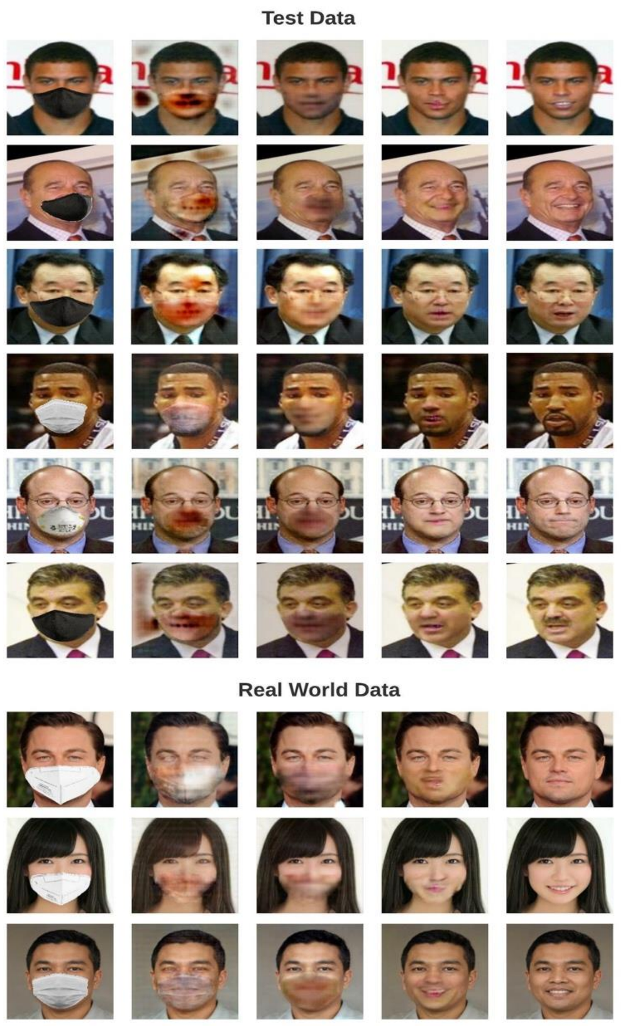

Figure 6: Result of three trains (left-to-right: input mask face, trainA output, trainB output, trainC output, original face)

Figure 6 shows multiple results generated by our diversified three training processes. From left to right, they are input mask faces, train A result (pix2pix), train B result (pix2pix+mask segmentation), train C result (customized GAN+mask segmentation), face without mask. Also, our result data is separated into two parts. The first part's input data come from our generation of synthetic dataset. We separate 100 face images to be our test data. As for the second part, we obtain several faces on the internet and use the same method to put the mask on these faces. The reason why we need this real-world data is that the Labeled Face in the Wild dataset contains similar resolution images. All three trained models tend to be familiar with this type of data. We need disparate data to differentiate these models.

There are two comparisons we can make based on our method. The first is between train A and train B. The difference between them is that we add a binary mask segmentation on our input data. Our initial hypothesis for the function of this additional channel is that it is conducive to both mask region inpainting and rest region generation. It is effective for inpainting the mask region since it guides the GAN to focus on restoring this missing region face. Meanwhile, our GAN might which results in modifying the rest of the face. Therefore, with the help of this focus, the quality of the rest face region can be improved as well since our model is more likely to "copy" the region other than mask instead of generating. The test result below verifies our hypothesis. The quality of both the rest face and mask region in our train B result are much better than our train A result. This contrast is even more stark in our real-world data. The resolution in our train B result is much higher, except that the mask region doesn't seem to improve a lot.pay attention to generate the region other than mask one if our input face images have no additional binary mask channel,

The second comparison is between train B and train C, where we adopt different GAN models to solve the face mask removal problem. The result shows that the effectiveness and quality of mask region inpainting improves a lot by using our customized GAN model. Our customized GAN model test data results are approaching the real face images. There are only minor problems in the lip region. As for the real-world data, the shape of nose and mouth become much clearer. Though they cannot match the quality of our test data results, it still indicates a substantial improvement compared to the Pix2Pix GAN model.

Our approach also has several limitations which can be improved later. First of all, the size of our training data might not be sufficient. It only contains 13k face images, which can be increased to a much larger dataset. With a larger dataset, our model can experience more diversity. Moreover, the resolution of the images in our dataset can be improved. According to our test on real-world data, our model is weaker when facing the high-resolution face images. Therefore, a high-resolution-image dataset can help overcome this weakness. Thirdly, a discriminator focusing on the mask region can be introduced to our model to improve our generative quality of missing mask region. Our problem boils down to only generating the missing mask region face. Our discriminator can also distinguish the images based on this part, which can be more effective to optimize the generator. Last, due to the complexity of training a GAN model, our training strategy can be modified as well. Two networks work in a GAN model, leading to a potential problem that our generator is likely to be too weak to compete with our discriminator at first. Training them all together might not produce an optimized result. Therefore, we can train our generator alone at the beginning of our training to enhance its competitiveness.

IV. CONCOLUSION

In this work, we have proposed 3 different methods to solve the facial mask removal problem. By comparing two different models, train A and train B, we can confirm that the additional binary mask segmentation is beneficial for both generation of rest face region and inpainting of missing mask region. This inspires me that if we can develop an algorithm to detect and capture the mask region in each masked face image, the effectiveness and quality of our facial mask removal will be improved a lot. Meanwhile, our proposed customized GAN model enhances the quality of mask region inpainting. It can also be extended to other missing region image inpainting problems.

REFERENCES

[1] Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative adversarial nets." Advances in neural information processing systems 27 (2014).

[2] Isola, Phillip, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. "Image-to-image translation with conditional adversarial networks." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1125-1134. 2017.

[3] Kwak, Hanock, and Byoung-Tak Zhang. "Ways of conditioning generative adversarial networks." arXiv preprint arXiv:1611.01455 (2016).

[4] Ronneberger, Olaf, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation." In International Conference on Medical image computing and computer-assisted intervention, pp. 234-241. Springer, Cham, 2015.

[5] Li, Ryan, Sreya Halder, and Divya Nagaraj. "Reconstructing temporally high resolution pediatric MRI scans from low resolution images."

[6] Din, Nizam Ud, Kamran Javed, Seho Bae, and Juneho Yi. "A novel GAN-based network for unmasking of masked face." IEEE Access 8 (2020): 44276-44287.

[7] Chen, Liang-Chieh, George Papandreou, Florian Schroff, and Hartwig Adam. "Rethinking atrous convolution for semantic image segmentation." arXiv preprint arXiv:1706.05587 (2017)

[8] Hu, Jie, Li Shen, and Gang Sun. "Squeeze-and-excitation networks." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 7132-7141. 2018.

[9] Hore, Alain, and Djemel Ziou. "Image quality metrics: PSNR vs. SSIM." In 2010 20th international conference on pattern recognition, pp. 2366-2369. IEEE, 2010.