Deepfake Generation and Detection Modeling

Clayton Bromley
CKB30@duke.edu

Namh Lahade
NSL14@duke.edu

James Qu
JQ46@duke.edu

Abstract

This paper explores different models for generating Deepfake images while also exploring how facial features can be manipulated to create convincing fake images. Through implementing two different models for facial generation as well as two different forms of facial manipulation, the paper compares and contrasts the different techniques used to generate fake faces through the use of GANs, diffusion models, and encoder-decoder architectures.

1 Introduction

Deepfake, or the act of manipulating image or video data to display a false reality, has become increasingly prevalent and realistic in the modern age. This technology provides a threat to the integrity of journalism, for now celebrities, politicians, and other people of interest can be shown in situations that never occurred. As such, there is an urgent need for robust methods to both detect and generate synthetic content given source image and video data. In this paper, a series of models will be explored to aid in the generation, manipulation, and classification of deepfaked facial images. Additionally defense mechanisms against deepfake attacks will be pursued. A list of explored deepfake models in this paper are as follows:

1. Artificial image binary classifier
2. Deepfake image generator using GAN architecture
3. Deepfake image generator using diffusion architecture
4. Facial image swap deepfake
5. Facial image reenactment deepfake

1.1 Motivation

The motivation for this paper is due to the wide applicability of Deepfake generation and classification. With misinformation being prevalent in many industries, working on a project to tackle the challenge of identifying misinformation was interesting. The dual nature of the project with regards to creating fake data and identifying whether data was real or fake was fascinating.

2 Models

2.1 Artificial image binary classifier

An essential part of the artificial image generation process as well as defense mechanisms against the hostile deployment of artificial images is the ability to accurately identify artificially generated images. GAN architectures, as described in section 2.2.1, partially rely on a discriminator model to determine if the generated images are realistic enough and training is complete. For the purposes

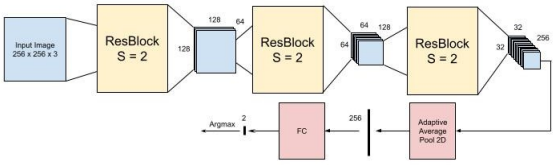

Figure 1: Artificial Image Classifier Architecture

of this paper, a residual neural network (RNN) model is trained to binary classify images as real or fake. The specific architecture of the RNN model used is shown in figure 1 and residual block architecture used is shown in figure 2. The data used to train this classifier was collected in two

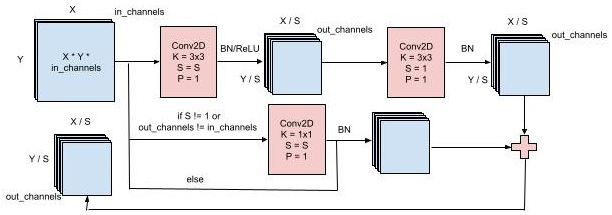

Figure 2: Residual Block Architecture

different ways. First, the real images used for training came from the CelebA dataset as described in section 2.2.2. The artificially generated images used to train this classifier originate from the diffusion image generation model described in the section 2.2.2. A total of 200 of each classification were used for training, testing, and validation with a 70/30 split for training and validation. The model was only trained for 13 epochs, as this was enough to achieve a maximum validation accuracy of 94.4% with a minimum validation loss of 0.589. The accuracy and loss curves are shown in figure 3. While this classifier is very good at classifying generated images based on the data given, there is

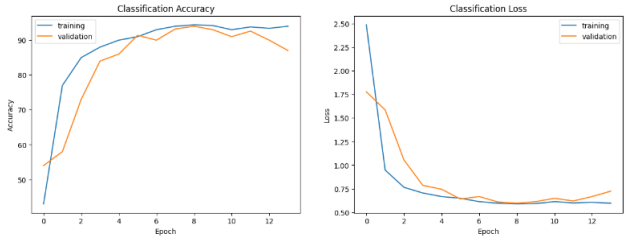

Figure 3: Training and Validation Loss and Accuracy for Artificial Image Classifier

very limited robustness. Because the generated data used is from the imperfect diffusion model, it is visually obvious that it is generated, as edges are very fuzzy. If more realistic generated face images were passed through the classifier, such as those generated by DALLE, then the accuracy would not be as high.

2.2 Deepfake image generator

Image generation is an essential part in the development of defense mechanisms against deepfake technology. Models such as the classifier previously described rely on generated data to determine what is real and what is fake. In this paper, two different image generation models are proposed and compared: a GAN model, and a diffusion model.

2.2.1 Generative Adversarial Network (GAN) model

GAN models are the state-of-the-art architecture used for artificial image generation. GAN architectures consist of two models, a generator that creates the artificial image given random noise, and a discriminator that determines the probability that the generated image is real. As the models train, the discriminator becomes an increasingly accurate classifier while the generator uses the

loss from the discriminator to make the generated images more realistic. At convergence, the discriminator outputs p(x is real) = 0.5, as the generated images and the real images are indistinguishable.

Both the generator and discriminator use a RNN architecture as shown in figure 4 and 5 respectively. The residual block structure is the same as the one used for the classifier and is shown in figure 2. Like other models in this paper, the GAN was trained using the CelebA dataset and produces images of size 256x256. Data augmentations including cropping, random horizontal flip, and random recolorations were used to improve model robustness. The pretrained output consisted

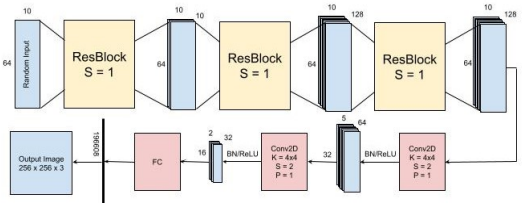

Figure 4: GAN Generator Architecture

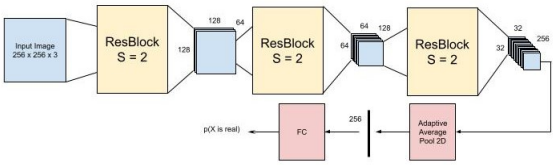

Figure 5: GAN Discriminator Architecture

of random noise, and as training progressed, features became emphasized. By 1900 epochs, the outline of a face became visible, and convergence occurred around 10,000 epochs. After convergence, oscillation occurs, so the final output is imperfect. Figure 6 shows a sample training image, the random output before training, the output after 1900 epochs, and a generated face after convergence. Figure 7 shows the discriminator and generator loss as well as the discriminator output p(x is real) for the first 280 epochs. The oscillations can be seen.

Figure 6: Output at Various Stages of GAN Training

2.2.2 Diffusion model

Diffusion models were also used to generate facial images. These models worked by going through a forward process to create noise and then using a backward process to denoise the images into new faces. For the project, the CelebA dataset from the torchvision.datasets package (which includes numerous samples of celebrity facial images) was used as inputs and preprocessing was done on these images through cropping and resizing the image to sizes of 256 by 256 and having a random probability of horizontally flipping the image.

The forward process for the diffusion model is used to add noise to the training images. This is done by using a Gaussian distribution to apply noise for each time step. Thus, as the image moves through each time step, it becomes less identifiable and more noise is generated. After many

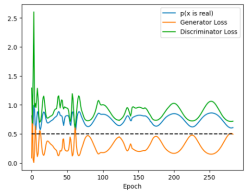

Figure 7: Discriminator Output, Generator Loss, and Discriminator Loss for First 280 Epochs of GAN Training

time steps have been applied, final images that were generated appeared to be completely random with an assortment of colors.

The backward process works by using a type of CNN called a U-Net. Overall, this part of the diffusion model uses probabilities to determine and predict the previous time steps and denoise the images. The U-Net works by using sine and cosine functions to calculate sinusoidal embeddings. These noisy images will then apply these embeddings for the specific time step in this backward process to help denoise the images. Upsampling and downsampling was then implemented to obtain the correct output channels. After incorporating the mean-squared error (MSE) loss, a new fake image was generated through this diffusion model process like the one shown in Figure 8.

After training the model on the CelebA dataset, noisy samples were obtained from this forward diffusion process. Then, the backward diffusion process was applied to denoise these noisy samples. These samples turned out to generate images that do appear to resemble very closely to a real human being. However, improvements could still be made as shown in section 2.1 where the classifier was able to detect real and fake images correctly with a high accuracy rate. Potential improvements can be done on these images by training for a longer amount of epochs and possibly increasing the amount of upsampling and downsampling to be done on the U-Net model to provide a more accurate model.

Figure 8: Fake Image Generated by Diffusion Model

2.3 Deepfake face swap

The main goal of the face swapping algorithm was to be able to swap James Qu's face onto Clayton Bromley's face. The main components of the face swapping algorithm are data loading and preprocessing, the encoder and decoder architecture, model training, and model application for face swapping. To create an effective dataset personalized to the group, 20 minutes videos of both James and Clayton were recorded, capturing various angles against a consistent background. These videos were processed into over 1000 images for each person using the OpenCV library.

Preprocessing was simplified due to uniformity of the image collection method, focusing mainly on data augmentation and normalization. The encoder used in the model comprised four convolutional layers with increasing filters and LeakyReLU activations, followed by dense layers and a PixelShuffler to enhance the quality of the data representation. The decoder largely mirrored the encoder's architecture but concluded with a sigmoid activation function for image generation.

The training involved two separate decoders for each individual's face, while a single encoder was trained on both datasets. This approach was designed to retain the unique facial characteristics of each subject in the face swap. An Adam optimizer with a low learning rate (3x10-5) was used over 300 epochs to ensure model convergence. The encoder-decoder pipeline is visualized in figure 9. The face swap pipeline is shown in figure 11.

To swap faces, an image of James was encoded into a latent representation and then reconstructed using Clayton's decoder. The model's effectiveness was assessed through visual examination of the swapped faces and monitoring the loss, which converged to 0.009. The image of the swapped faces can be seen in Figure 10. The final image has a lot of blur, there are 2 sets of eyes and 2 sets of noses, and the image is slightly pixelated. However the model does add parts of James' face onto Clayton.

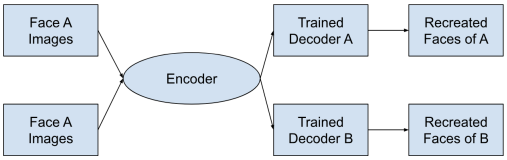

Figure 9: Fake Generated by Encoder Decoder Model

Figure 10: Image Generated by Face Swap Model

Figure 11: Pipeline to Use Face Swap Model

2.4 Deepfake face reenactment

Face reenactment modeling relies on GAN architecture. The face reenactment explored in this paper was created with the goal of applying deepfake to an image of James so it matched the expression of an image of Namh. Although the architecture of the model was fundamentally similar to that used in face generation as described in section 2.2.1, there was a notable distinction in its initial approach. Instead of beginning with random noise, the model initiated with an existing image specifically targeting James. This method ensured the preservation of James' distinct facial features during the reenactment process.

For training, the GAN model was exposed to a dataset composed of Clayton, James, and Namh. This dataset was created using the same method of developing datasets for the face swap algorithm. A key characteristic of this dataset was its uniformity in terms of lighting conditions and the centered alignment of the faces, which was crucial in enabling the model to focus primarily on learning a wide array of facial expressions while maintaining a standard environmental setup. The model was converged on 2000 epochs and the final images of face reenactment can be seen in Figure 12. As we can see, the final image is blurred with noise scattered throughout. However the expression of Namh was translated onto James' face.

3 Discussion

For the face generation models in section 2.2, fake images were generated that resembled human faces as seen in the images for those sections. By comparing the results from the two models, this project was able to compare the advantages and disadvantages of them. Diffusion models appeared to generate slightly clearer images for this project while GAN models proved to be incredibly compute-heavy with convergence only occurring after around 10000 epochs. As seen through the bottom left image in Figure 6, the image somewhat resembles a face with the eyes and mouth slightly

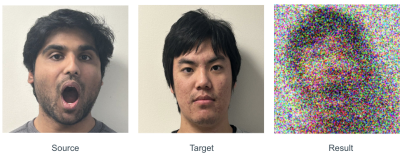

Figure 12: Image Generated by Face Reenactment Model

visible. However, the image still demonstrates that it is nowhere close to convergence after running for 1900 epochs. While diffusion models do have an advantage in being more time-efficient than GAN models, they do lack the discriminator that the GAN models have to identify real and fake images. Instead they rely on probabilities and likelihoods instead. Thus, these weaknesses make it so that in a setting where there are very large datasets and the training contains no time limits, the GAN models will eventually be able to perform better than diffusion models.

For the face swap algorithm in section 2.3, the combined image was blurred and pixelated. Also, features on the image like the eyes and nose were duplicated and some weren't in the completely correct place. However, the model did make a reasonable attempt at swapping James' face onto Clayton. This could be due to the fact that the model didn't run until full convergence. This could also be due to the fact that the dataset wasn't rich or diverse enough. To improve the model, the current could also be combined with public datasets. The face reenactment algorithm in section 2.4 also had a final image that was blurred and full of noise. However, Namh's expression was translated onto James in the output. Reasons for why the images weren't clear are similar to the face swap model as the dataset was also custom made.

3.1 Defense mechanisms

Deepfake images could be harmful to society and defense mechanisms are thus needed to prevent misidentification of the right or wrong images. Because these images seem to be a pretty good representation of a human face, sometimes a blind eye test cannot identify the results of which images are fake and which are real. Thus, different methods involving multi-pronged approaches have been used to try to defend against the malicious activities that Deepfake image generation could result in.

One common defense mechanism uses an authentication process to ensure that it can identify which images are real and which are fake. This process works by placing a watermark on the real image. Thus, when the Deepfake image is generated, it will not have the same watermark as that of the real image. Then, when authentication is used to identify whether the watermark exists in the image or not, it will look for the real watermarks.

Other features to look out for in fake images and help identify them better include observing many subtle aspects of faces to determine the unusual placement and occurrence of facial features. These can include the scaling of different aspects of a face such as the mouth, nose, and eyes compared to the rest of the face. Another aspect to inspect for in these Deepfake images is to check the lighting and the shadows that could appear in these images. Inconsistent lighting often occurs in Deepfake images, and these can be a dead giveaway for identifying the differences between which images are real and which are fake. Thus, there are many other complex algorithms that detect these certain elements in the lighting and facial features to better classify real and fake images. An example of this model trained to identify these inconsistencies was used in the RNN referenced in section 2.1. As described in the section there, this classifier performed its task well as it was able identify the fake faces that were generated through this project.

4 Conclusion

This project was able to analyze the intricacies of image transformation of facial features including face generation, face swapping, and face reenactment as well as ways to defend against the malicious uses of Deep Fake images.

References

[1] Bradski, G. (2000). The OpenCV Library. Dr. Dobb's Journal of Software Tools.

[2] Cheigh, Justin. "Generating Images Using Vaes, Gans, and Diffusion Models." Medium, Towards Data Science, 7 June 2023, towardsdatascience.com/generating-images-using-vaes-gans-and-diffusion-models-48963ddeb2b2.

[3] Chen, Yunzhuo, et al. "Text-Image Guided Diffusion Model for Generating Deepfake Celebrity Interaction." Arxiv.Org, 26 Sept. 2023, arxiv.org/pdf/2309.14751.pdf.

[4] Ding, Xinyi, et al. "Swapped Face Detection Using Deep Learning and Subjective Assessment - EURASIP Journal on Information Security." SpringerOpen, Springer International Publishing, 19 May 2020, jis-eurasipjournals.springeropen.com/articles/10.1186/s13635-020-00109-8.

[5] Large-scale CelebFaces Attributes (CelebA) (2015). [Dataset]. Proceedings of International Conference on Computer Vision (ICCV). https://mmlab.ie.cuhk.edu.hk/projects/CelebA.html

[6] Noreen, Iram et al. "Deepfake attack prevention using steganography GANs." PeerJ. Computer science vol. 8 e1125. 20 Oct. 2022, doi:10.7717/peerj-cs.1125

[7] Tripathy, Soumya, et al. "Facegan: Facial Attribute Controllable Reenactment Gan." arXiv.Org, 9 Nov. 2020, arxiv.org/abs/2011.04439.