

Progress Report

List of Group Members: yuxuanz6 (Yuxuan Zhou) klfeng2(Keven Feng)

An updated statement of the project definition and goals

Description: Recently people have implemented the ability to create synthetic videos of a person saying things they never really said. This technology is known as Deepfake which could be done by changing the real person's face into a fake face. We would like to implement this technology and compile it into an application which can be used by individuals for personal use. In addition, we would also like to implement a detector which can be used to find fake videos to try and protect against malicious use of this software. To achieve this, a machine learning algorithm structured around the GAN model will be used. The GAN model has two neural networks constantly competing with each other to optimize their results. The neural network called the generator generates fake images. The neural network called the discriminator determines whether or not an image is fake or real. By using the GAN model we will inherently build two networks that can perform image faking and fake image detection.

Goals:

Min (Changed): Application that can swap the face in two images of different people

Max (Changed): Application that can swap the face in two videos of different people

Bonus (Changed): Face swap application that can swap your face with any of a set of predefined celebrity faces.

Reason for changes: When training the models we realized that the training time is too long to train in real time. Thus if we predefine the set of people you can face swap too, we can just use pretrained models to perform the face swap in real time.

Current member roles and collaboration strategy

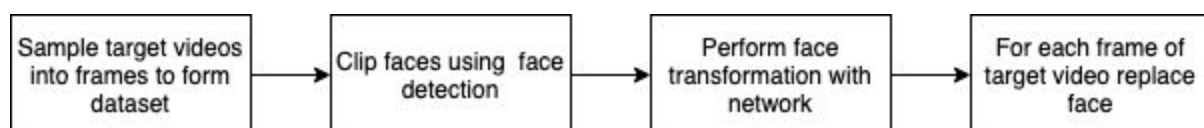
Yuxuan Zhou is responsible for the face detection code that cuts out the face from source image or video and the face replacement code that pastes the transformed face back into the source image or video. Also responsible for GANs model optimization.

Keven Feng is responsible for the machine learning code that transforms a source image face into a target image face. Also responsible for investigating research papers. Both members are responsible for writing the research paper.

The code is shared by google drive and across Slack. We meet at least once a week from 1-4 pm and have online discussion via Slack. During the meeting, we exchange our ideas and write the report, make some project changes accordingly.

Proposed approach

VideoProcessor: The video processor has to handle the transformation of video to images and images to videos process. To do this a face detector will be used to gain only faces images which will be used to form our training dataset. We will also use the face detector to gather the faces we need to transform once our model has been sufficiently trained. Currently we obtain our frame samples using the preprocess_video() function from the preprocess library in <https://github.com/shaoanlu/faceswap-GAN.git>. We sample at a rate of every frame, although by default it is every fifth frame. Below we illustrate the flow of our design.

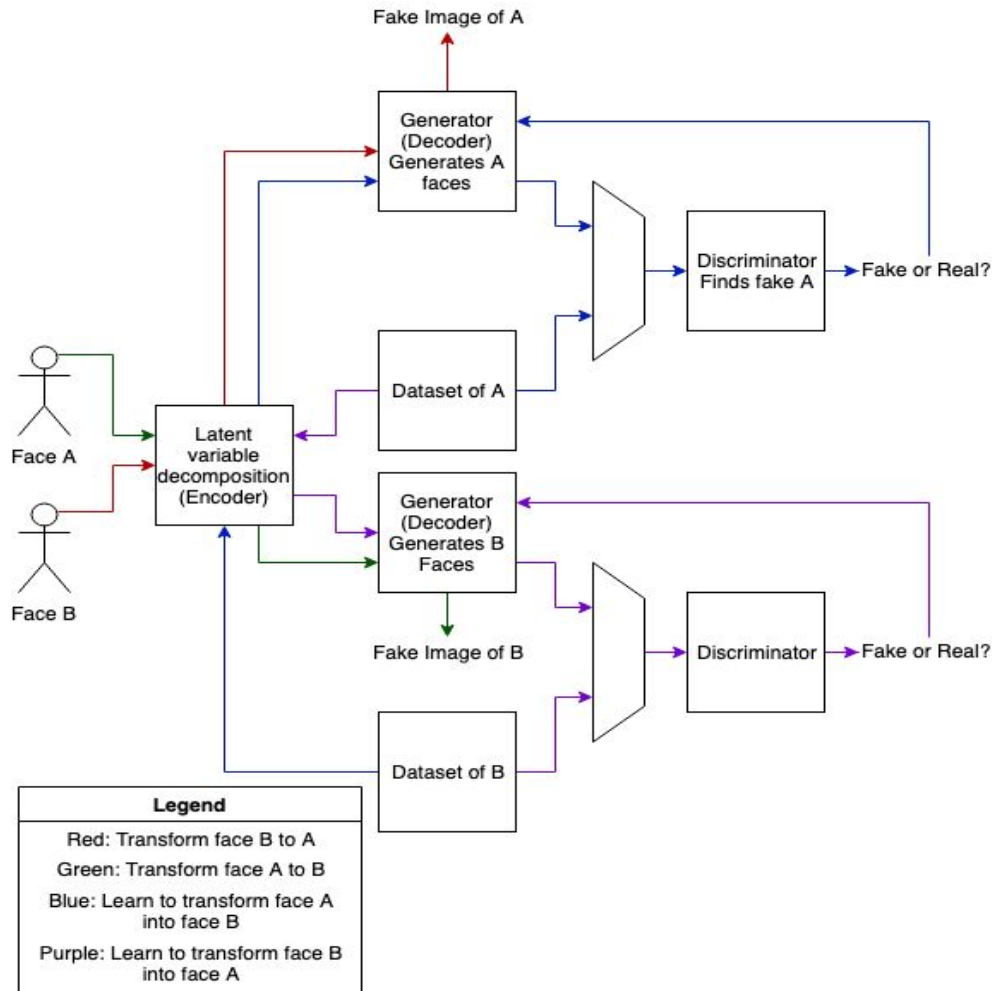


Progress Report

MTCNNFaceDetector: For face detection we choose to use the MTCNN face detector. The reasons here are its flexible usage and code refactoring. We could change the resolution accordingly and delete unsuccessfully by using the MTCNN face detection etc. It's implemented with class (https://github.com/kpzhang93/MTCNN_face_detection_alignment), return pointer: `fd = MTCNNFaceDetector(sess=K.get_session(), model_path="./mtcnn_weights/")`. We could change the key points to be standard 68 points or whatever is in the reason range, here we use 100 points. Note, this class has several functions here and many are from outside resource.

Face transformation:

Below is a diagram which represents the face transformation layout for two distinct people.

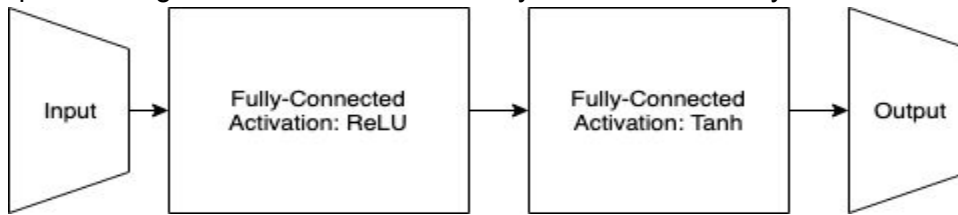


Encoder: From the GAN paper, (<https://arxiv.org/pdf/1406.2661.pdf>) GAN's generate new convincing images from noise. However for this problem we need to preserve the facial expressions across transformation. To do this, instead of feeding in random noise, we implement an encoder network that tries to learn a common latent space among both targets to preserve facial expressions. If we learned on solely random noise, we have no guarantee that the latent variables Generator A and B use would be the same.

GAN: Currently our GAN implementation is based off this source implementation https://github.com/mchablani/deep-learning/blob/master/gan_mnist/Intro_to_GANs_Exercises.ipynb, however we plan to investigate more complex networks that match this network https://github.com/shaoanlu/faceswap-GAN/blob/master/networks/faceswap_gan_model.py.

Progress Report

Network: Currently each network is designed with the same layers and the only difference is the size of these layers. We have two fully-connected layers with the first layer having a ReLU activation function and the second layer have a tanh activation function. We plan on experimenting with different number of layers and kinds of layers like convolution.



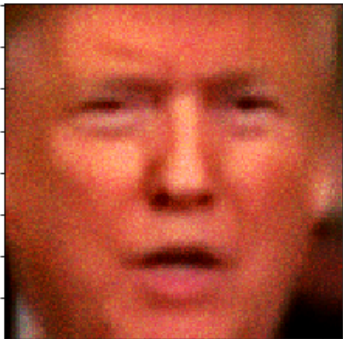
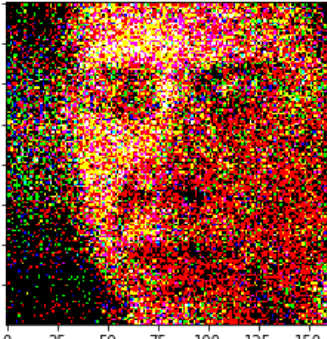
Data

Cage/Trump dataset Example:

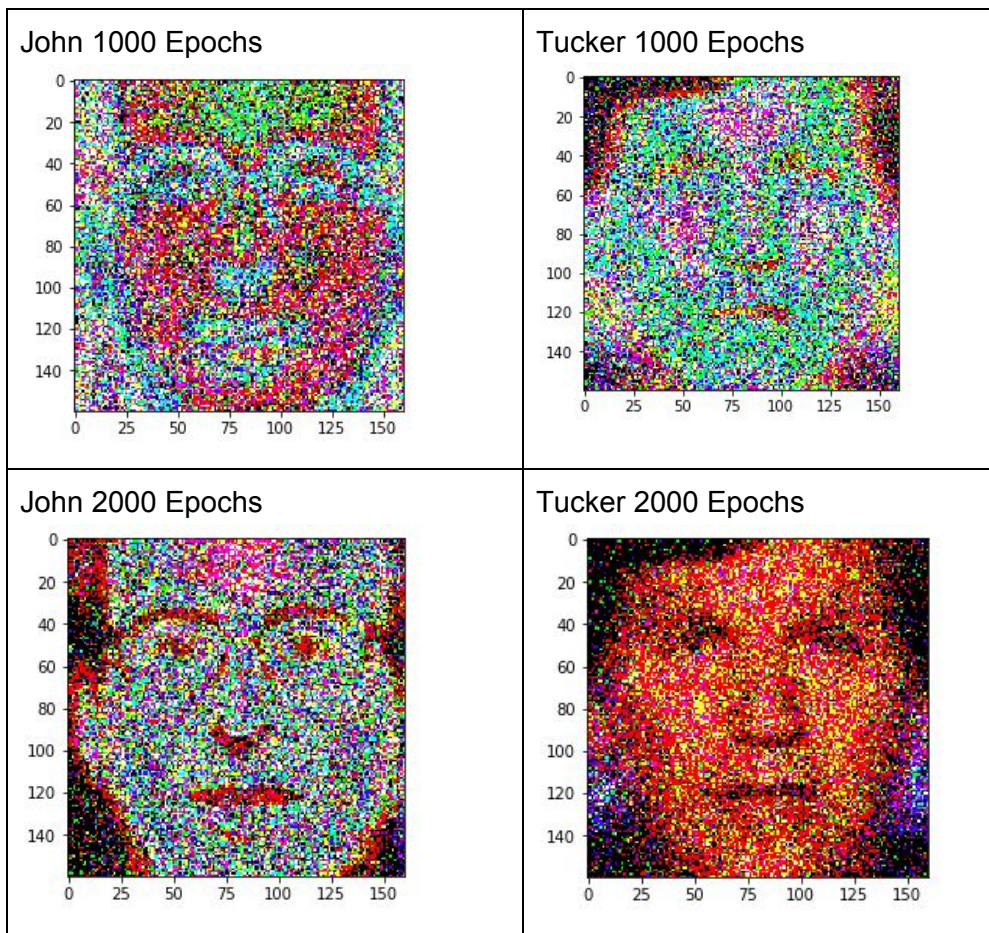
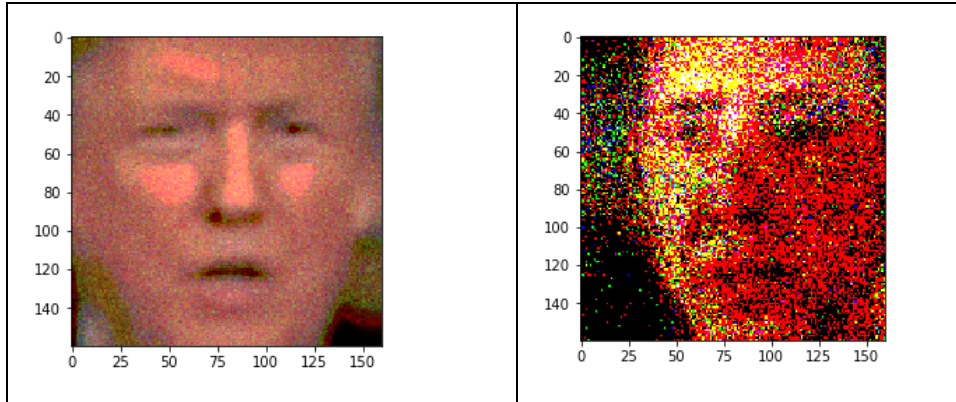
person A's video: trump.mp4 (first 50 second download using python script by youtube)	Person A's images after Face A preparation step, totally get 3533 faces, image size will be reshape to (160,160,3)	Take the first 1000 A faces for training and set the batch size to be 100 and epoch to be 200 for one time, totally need to be ~20000
person B's video: cage.mp4 (first 50 second download using python script by youtube)	Person B's images after Face B preparation step,totally get 3533 faces, image size will be reshape to (160,160,3)	Take the first 1000 A faces for training and set the batch size to be 100 and epoch to be 200 for one time, totally need to be ~20000

Initial results

Currently we have the face transform network as described above implemented. Right now our results only show noisy original images and not transformed images. Looking at our latent space we observe that the encoder is currently outputting the same latent space, so either our datasets need wider variance or we need a better network.

Trump's face A after 200 epoch 	Cage's face B after 200 epoch 
Trump's face A after 1000 epoch	Cage's face B after 1000 epoch

Progress Report



Current reservations and questions

The main issue we are running into right now is that the training time takes too long. From the reference project we looked at, it was suggested that around 27k training iterations should be done, but we calculated that this training would take 9 days just to implement a faceswap GAN that can swap only 2 given people. When we added GPU capabilities the training time is now estimated to take 3 days, but it is still a long time to train before results can be verified. In addition to training, we are unsure if the neural network model we chose can accurately represent the problem. Also tuning the hyperparameters will be hard with such a large training time.