Submitted by: Dhruv Desai, Rahul Shekhar

1. **Hyperlink**
   Github Repository[1]

2. **Introduction**
   Conditional GANs work as an extension to the GAN framework as many times we want to generate data based on a particular condition/label. For the scope of this project, that condition is age. Our end goal is to be able to take a person's face image and then be able to generate how they would look at a different age. Our inspiration for this project is taken from *Face Aging with Conditional Generative Adversarial Networks.*[2]

   To accomplish this task we use a subset of an existing dataset and choose two categories of age and assign labels to them. We run classification models on that subset to support the claim of the inductive biases. Facial features of humans change as they grow older and the model tries to learn how a person looks at the given age group. The Generator learns to condition on an image based on the label and produces an image based on it.

3. **Dataset**
   For the scope of this project we used a subset of IMDB-WIKI 500k+ face images with age and gender labels[3]. Images were filtered based on the following age group 20-30 year old and 40-50 year old. They were assigned labels zero and one respectively. The final dataset had 13806 images where 7000 images were assigned to label zero and 6806 images assigned to label one.
   The above calculations and splits were made using the original *wiki.mat*[3] (metadata file for the original dataset), mat_file_editor.py in the github repository contains the code for label assignment and age calculation. wiki_edit.mat contains the metadata which will be used for the rest of this project.

Figure 1: Sampled images from dataset with labels

4. **Data Hypothesis**
   For this dataset we have the hypothesis that people of different age groups have different have different features with respect to their faces. The scope of this project is limited to two age

groups 20-30 years and 40-50 years.

5. **Architecture Priors**

   In this project we use a logistic regression model, a convolution neural network architecture and a conditional GAN architecture. In case of the CNN which are transnational invariant. For the GAN we initialize the weights using a normal distribution with a given mean and standard deviation.

6. **Non-Deep Learning benchmarks**

   On the basis of the data hypothesis and priors we ran a logistic regression model to classify the two different age groups. We used 9120 images to train the model and the remaining 4704 images were used to test the model. The main intuition behind running this classification model was to support the claim of the inductive bias that people of different age groups have different features. The final training and test accuracy for this model were 65% and 66% respectively.

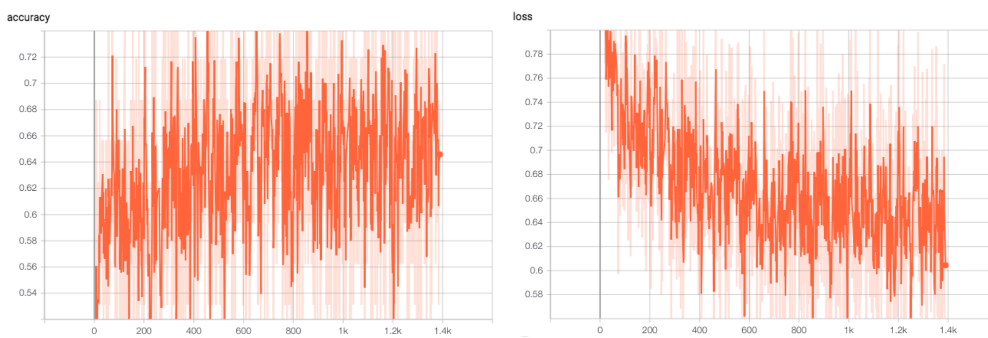


Figure 2: Training curve and loss for logistic regression

7. Base Deep Learning model: Similar to the non-deep learning base model, we ran a CNN based architecture for this model. This model had four convolution layers with batch-normalization after each layer. The final training and test accuracy for this model were 66% and 69%.

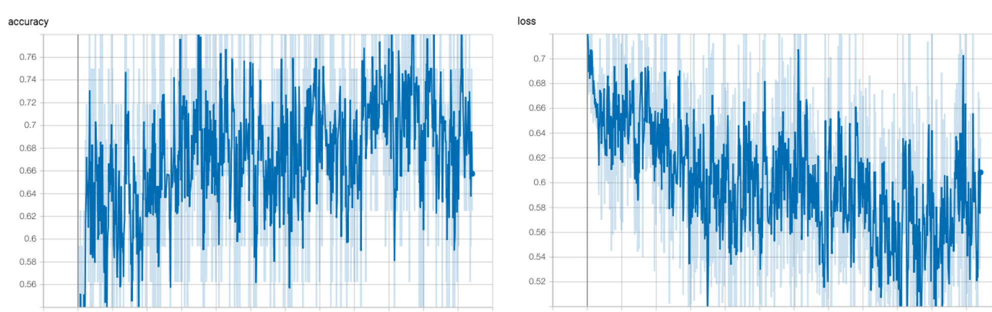


Figure 3: Training curve and loss for CNN

8. **Advanced Deep Learning Model**

   Using this dataset we build a conditional DCGAN (cGAN) to generate images of people in these age groups. A conditional GAN works similar to a DCGAN but in addition to the images, the labels are used. The generator learns to generate images conditioned on the label of the image. Once the cGAN is trained, we build an encoder that learns mappings from the outputs of the Generator to their respective noise vectors. This allows us to generate a noise vector for any input image, add an age condition of our choosing and then pass this vector through the Generator to get our desired output.

Adding Conditions to Input Vectors

The labels are padded to the input noise vector as one hot encoded vectors and fed to the generator. For the discriminator, we pad one hot vectors using the label of the input image after the first convolution layer with size equal to the size of the image at that layer.

cGAN Architecture

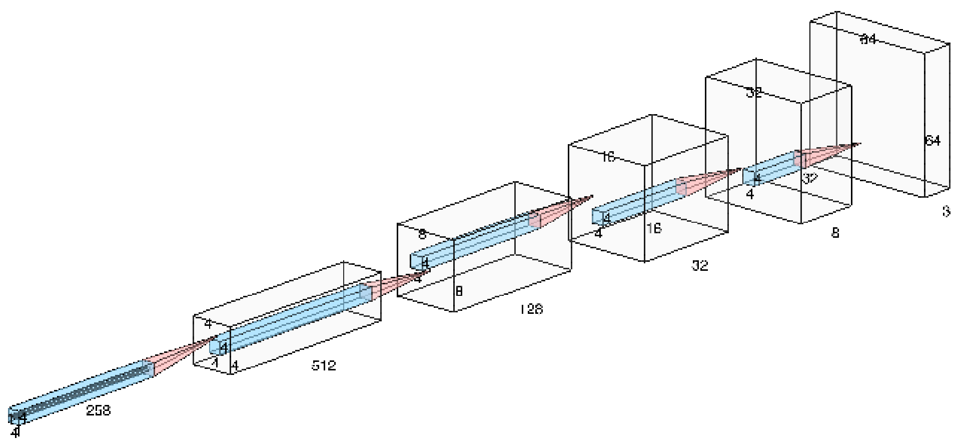


Figure 4: Generator Architecture

For the Generator architecture, we use ConvTranspose2d layers with a filter size of 4 along with BatchNorm and ReLU activation functions. For the last layer, we remove the BatchNorm and use a Tanh activation function.

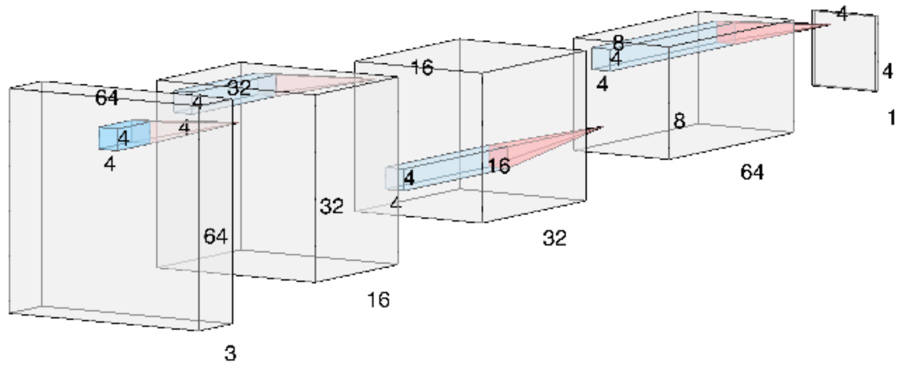


Figure 5: Discriminator Architecture

For the Discriminator architecture, we use Conv2D layers with a filter size of 4 along with Batch Norm and ReLU activation functions. For both the first and layer, BatchNorm is removed and a Sigmoid activation function is used in the layer.

Encoder Training and Architecture

In order to get training Data for the encoder, we generate noise vectors z_i along with their age labels y_i and then pass it through the Generator to get output image o_i i.e $o_i = G(z_i, y_i)$. The encoder is then used to learn a mapping from o_i to z_i. We generate 100K such examples and use it as the training data for the encoder.

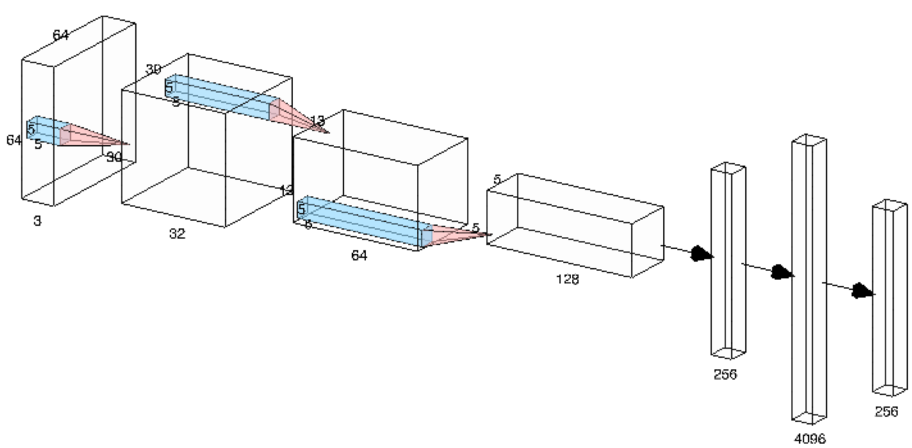


Figure 6: Encoder Architecture

The Encoder uses filter sizes of size 5 along with BatchNorm and ReLU activation functions. No max pool layers are added. In order to make sure the encoder is able to learn the distrbiution of the noise vector which is $N(0, 1)$, we do not add an activation function to the last fully connected layer.

Hyper parameters used for Training

	Generator	Discriminator	Encoder
Learning Rate	0.001	0.0005	0.0002
Optimizer	Adam	Adam	Adam
Epochs	85	85	25
Batch size	64	64	64
Latent Noise Vector	256	-	-
Loss Function	BCE	BCE	MSE

Table: Hyper parameters used for training

GAN Visualizations and Results

We trained the GAN for 19,000 iterations. Following are the plot for loss over epochs while training the cGAN, the generator starts oscillating after 12k iterations and around the same iteration number the discriminator converges.

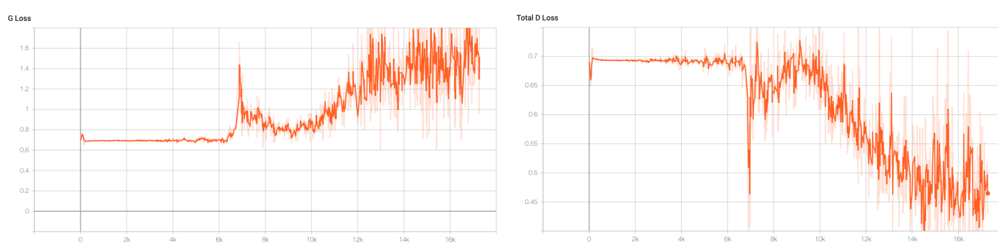
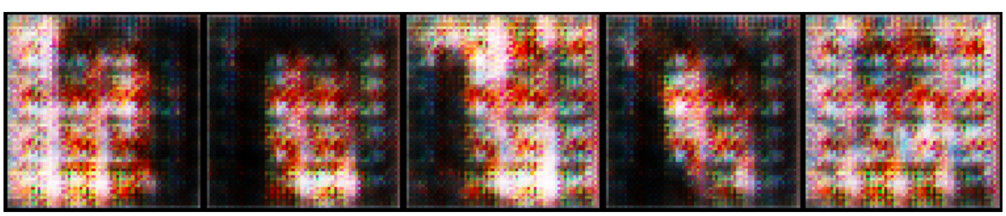
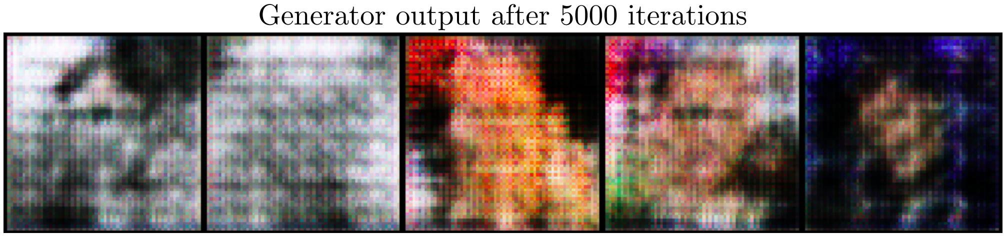


Figure 7: Generator and Discriminator Loss

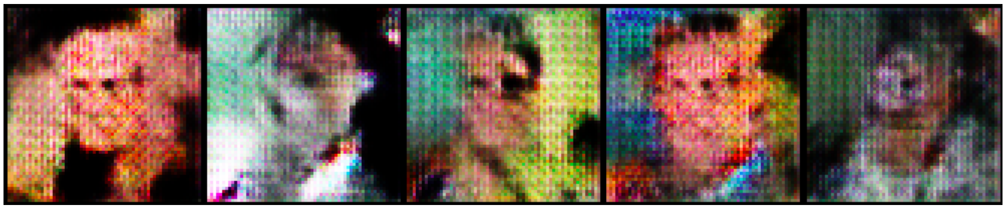
Following are the GAN output images taken during different iterations during training,



Generator output after 5000 iterations

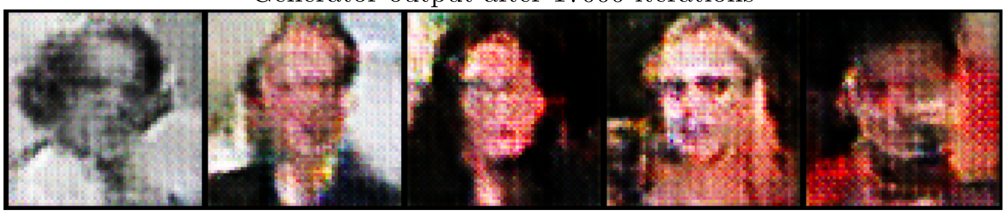


Generator output after 10000 iterations



Generator output after 12000 iterations

Generator output after 17000 iterations

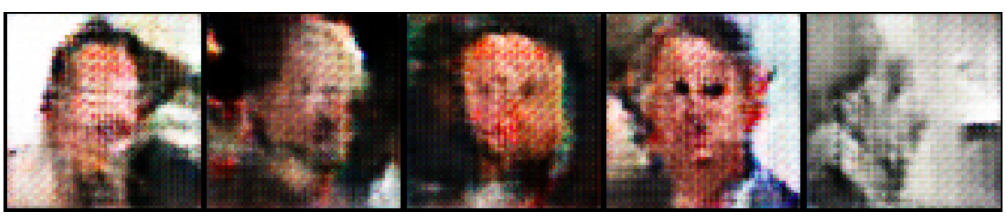


Generator output after 19000 iterations

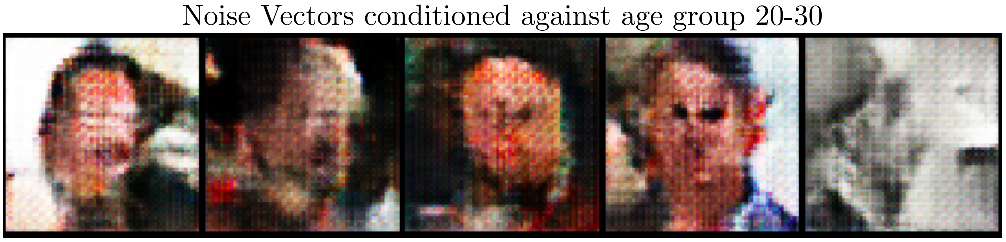
Figure 8: Generated Images while training cGAN

The above images depict how our GAN becomes better at generating images.

We would also like to see how well our GAN does in generating face images at different ages. In order to test that, we generate noise vectors and run it through the Generator once with a label of 0 and once with a label of 1. Below are the outputs of this experiment.



Noise Vectors conditioned against age group 20-30



Noise Vectors conditioned against age group 40-50

Figure 9: Conditioned Outputs

Clearly, there isn't much difference between the two. We think the two mains reasons for this are:

- There isn't that drastic a change in the features between people of ages 20-30 and 40-50.
- The outputs of the GAN are blurry due to which its hard to make out what the actual facial features are.

Encoder Visualizations and Results

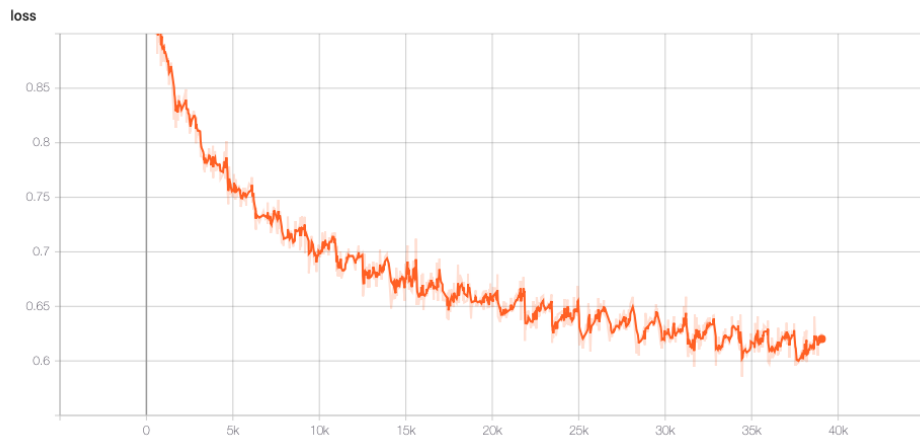
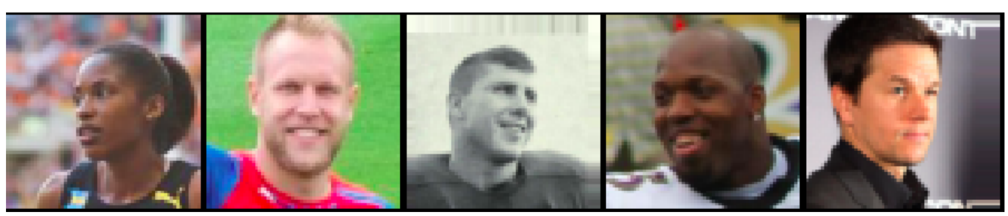
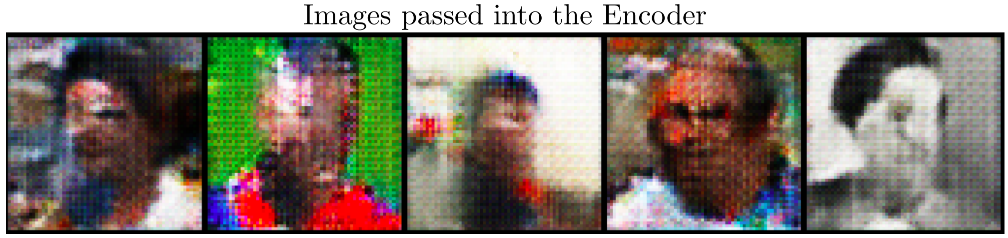


Figure 10: Loss over epochs for Encoder

The encoder converges to a loss of about 0.6. This loss is important because it represents the information we lose during the mapping process. Based on the loss value, we lose about 50-60% of information.



Images passed into the Encoder



Output of Encoder passed into the Generator

Figure 11: Final images passed from Encoder to Generator

The first image is the input to the Encoder. The output of the Encoder is fed into the Generator, the outputs of which is depicted in the second image. Based on the second output, we can see that our Encoder was successful in learning a mapping between the cGAN outputs and the noise vectors. We can see the structure, pose and certain features of the image is learned by the encoder. However, we can see that these outputs are more blurry as compared to the Generator outputs. This is when the loss of the Encoder becomes important. The 50-60% of the information that we lose is in the sharpness of the image, due to which these results are more blurry.

Improvements

- Based on the results of the cGAN, we saw that there seems to be very little difference between most people of ages between 20-30 and 40-50. Adding more age groups as well older age groups should help the Generator in learning more discerning features between them.
- We can see that we lose information in mapping the generator outputs to their respective noise vectors. Optimization techniques can be used to reduce this loss.

References

[1] Rahul Shekhar Dhruv Desai. Cis700 final project github repository, 2019.

[2] Grigory Antipov, Moez Baccouche, and Jean-Luc Dugelay. Face aging with conditional generative adversarial networks. *CoRR*, abs/1702.01983, 2017.

[3] Rasmus Rothe, Radu Timofte, and Luc Van Gool. Deep expectation of real and apparent age from a single image without facial landmarks. *International Journal of Computer Vision (IJCV)*, July 2016.