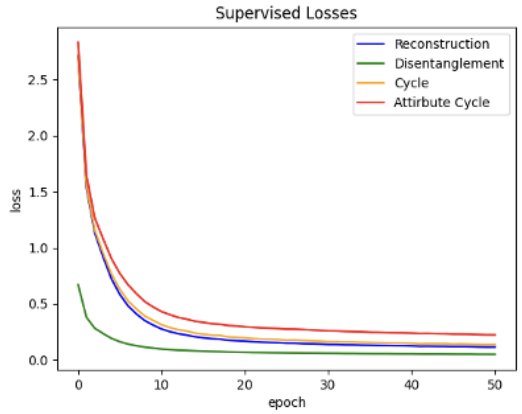
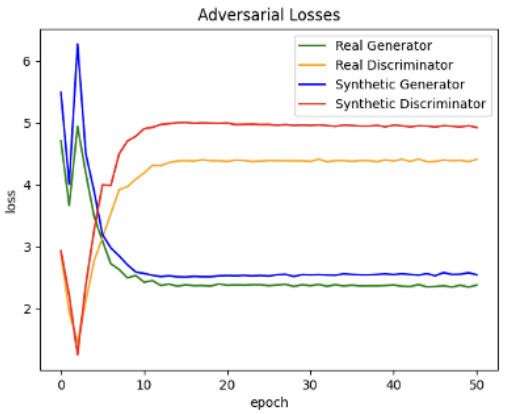
Topic: PuppetGAN
Team member: Yuan Zhang, You Peng, Junyi Zhu

# Benchmark

Github repo: [https://github.com/GiorgosKarantonis/PuppetGAN](https://github.com/GiorgosKarantonis/PuppetGAN)

Use Mnist dataset, losses during 50 epochs:




After 50 epochs train, the results shown as below:

# Data

We used two databases for this project: Weizmann and SynAction dataset.

The Weizmann dataset contains real human action videos, and the data is in an AVI format. The data contains a total of 10 actions, each of which is performed by 9 testers, and all of them use the same background. The SynAction dataset is a large synthetic human action dataset
provided by the TwoStreamVAN team. The data format is a GIF (Graphics Interchange Format), which contains

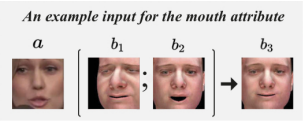
train/test_real

train_synth_triplet

10 kinds of actions, and each action is done by 10 3D characters, and each action of each 3D character has 14 different angles to provide a 3D perspective.

## Preparing and processing data

We convert each frame of the avi and gif format files of these two datasets into png pictures, and process the pictures as the input of the model. We need to prepare 4 kinds of data, which are the real and the synth for training, and the real and the synth for testing.




Example from PuppetGAN paper

In the real part of the training data, we used each frame in the video provided by the Weizmann dataset to become a 128*128 png image. Using 4999 pictures  can get a good result as mentioned in the puppetGAN paper, so the first 4999 pictures are selected as the data of the real part of the training set.
The part of training synth is the triplets pictures, of which from top to bottom correspond to the b1, b2, and b3 pictures in the puppetGAN paper. And before training, we need to fill the right side of the triplets pictures with white blank, so that the length and width of the picture become 128*3=384 pixels.

train_synth_padded

The real part of the testing data is processed in the same way as the real part of the training data. In the synth part of the testing data, we extract one frame every 5 frames of the gif from the SynAction dataset, and take a total of 10 images, and finally splice them together to form a long image.

# Change code

config.json:

```
"bodies" : {
      "info" : "dataset-specific hyperparameters",

      "batch size" : 30,
      "image size" : [128, 128],
      "save images every" : 10,
      "save model every" : 10
  }
```

# Experiments

We tried 3 different sizes of input, 32*32, 64*64, 128*128.

## Observations during training

### Training time

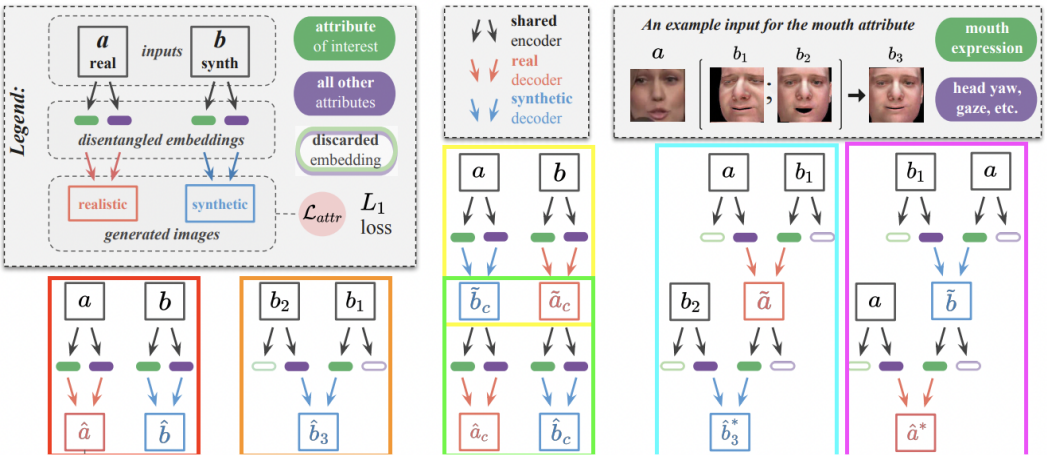
The model takes 5, 7.5, and 15 hours, respectively when training on 32x32, 64x64 and 128x128 images. While the image size grows exponentially, time cost does not, indicating the model is capable of dealing with large size images for the time end.

### Predicted result of training set

epoch=0 epoch=50 epoch=100

Above are the results in the training set, where epoch = 0, 50, 100, respectively.

Firstly, explain the above pictures: the first line is input images, from left to right are the pictures corresponding to a, b1, b2, and b3 mentioned in PuppetGAN paper (as you can see it below). The remaining 2 to 7 rows correspond to the results in the 6 different color grids from left to right and top to bottom in the figure below.



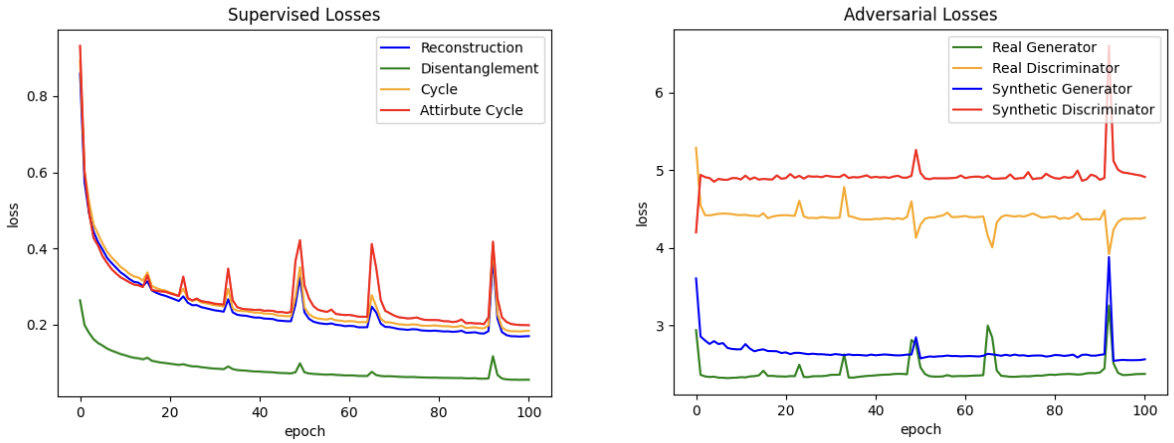

From PuppetGAN paper

As epochs grow, the model's performance gets a prominent increase in training results. Also, the model learns better about what the attribute of interest to extract from the real human image and which part to remain. Then, the model learns what human contour and human pose are. And the reconstructed image gets a higher resolution.

Above is the result when epoch = 10. We find the model almost achieves the goal at this time though the result is in a low resolution. Actually, this indicates that the higher resolution the image is (in other words, the construction becomes more complex), the more epochs the model needs, which corresponds to the training time the model takes on different image size data.

# Result

Our final model uses the images with size = 128*128 as input and trains for 100 epochs. The losses record during the training process shown as below:




From the above figure we observe:

1. The training results of Real and Synthetic Generators are good, and the same conclusion can be obtained from the image results generated during the training process.
2. In general, the Generator and Discriminator losses of the Real part are smaller than those of the Synthetic part. But because from the pictures generated during the training process, we can observe that the result of Synthetic part is actually better, so we analyze the reason that the loss of the Real part is lower than the Synthetic part is that the characters in the Real part images are relatively smaller and the background is almost the same, so the losses will tend to be small.
3. After 10 epochs, the learning efficiency dropped drastically, accompanied by drastic fluctuations.

## Result Observations

We believe that the size of the characters in the two data is too different. The characters in the training real data are much smaller than the characters in the training synth data, so it is difficult for the model to learn the attributes of the people in the training real data. Through observation, the model learns the background in the training real data very well, since you can see that there is a window in the upper right corner all the time. From the picture below, it is the result after 100 epochs, we can observe that the predicted character is always in the same place, which is

always in the middle position of the image. We consider this is because the person in the synthetic data image is in the center, so the predicted result also moves the character to the center position and does the same action.

Limited to the low resolution of real human part input data, the characters in the image are not clear originally, the characters in the image are too small, which leads to insufficient information about the attributes of the characters, and we only trained for 100 epochs. As can be seen in the above figure, in the final result, the image of the character is rather vague, and the "foot" did not leave the ground during the whole jump, which is not “correct”.

What’s more, many results show almost the same predicted image even though the input data are totally different. Sometimes, the model tries to simulate the outline of the frontal movement with the sideways movement. In other words, the final figure is shown in sideways form. Only the contour is synthesized, but the posture is not.

From looking at the gif-file (gif files can be found in the folder ./results/test/epoch=100/, there are also test results at other epochs in the ./results/test/ folder), when the images are continuous, you can actually observe the characters look like they are jumping up, which shows that the model does have learned the movement attributes of 3D characters.