In this paper, the authors presented a new solution to learn interpretable and meaningful representations in Generative Adversarial Network (GAN). Given the time and GPU hardware availability, I have used the simplest datasets; MNIST and Fashion MNIST. As the paper explains about the architecture in a detail, I followed exactly the same setup provided in Appendix C.1. MNIST section of the paper. Experiment 1 and 2 are divided simply in the order, not representing which represent which. I thought that comparison with original GAN and manipulating latent code both were demonstrating the key concepts (disentanglement representation and maximizing mutual information) and the performance simultaneously.

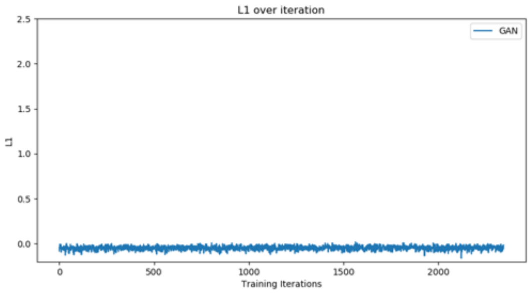
Experiment 1: Comparison with original GAN on MNIST

To compare with original GAN, I compared the output performance between original vanilla GAN and infoGAN. Quantitatively, I checked the L1 values for two networks and qualitatively, I compared the final generated output images. If you see the generated image on the left, it its clearly shown that the infoGAN's generator produced much more neat outputs than simple Vanilla GAN. Especially, the random noises are not present in the infoGAN images while Vanilla GAN is disturbed by many small noises. It is not concrete to state this, but it could be due to the properly learned mutual information.

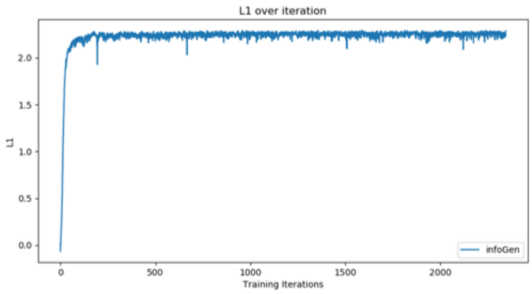
Vanilla GAN

infoGAN

If you see the figures below about the lower bound L1 over the iteration, the implementation follows the paper demonstration; while InfoGAN lower bound L1 is maximized to ~2.3 in less than 500 iterations, the Vanilla GAN's lower bound L1 even struggles to reach the value above 1. Therefore, this indicates that my implemented network did show mutual information maximization between latent codes c and generated images.



Vanilla GAN



infoGAN

Experiment 2: Manipulating Latent Codes

If you see the image below, it represents categorical code (c1) on each row and continuous code (c2, c3) on each column. Very similar to the paper produced results, a small value of c2 denotes left leaning digit whereas a high value corresponds to right leaning digit. c3 was very similar to the paper results as well; smaller c3 shows shallow digit while a bigger c3 shows fatter (flatten) digit. For the Fashion MNIST data, there was some changes over varying c3 from -2 to 2; for instance, the shoes type at the last row changed from high-hills to normal sneakers and in the fifth row, a long t-shirt changed to a duffle bag. This demonstrates that the network did learn something in latent codes and showing its effects; however, it is somewhat inconsistent.

Categorical (c1)

Rotation (c2)

Width (c3)

Width (c3)

Limitations and Discussion

- Perhaps the most important limitation of current implementation is that I only utilized simple dataset such as MNIST or FashionMNIST. However, this was inevitable given the hardware availability and hope to be understood with generosity. Also, FashionMNIST result showed somewhat confusing results regarding the latent space.