

# **CmpE 597: Homework 2-Report**

**Name:** B. Can Koban

**Student No:** 2021700096

**Mail:** burakcan.koban@boun.edu.tr

Department of Computer Engineering  
Bogazici University  
Istanbul, Turkey, 34342

## 1. Question 1

### 1.1. Model Summary

In this section, I briefly summarized the model architecture, output dimensions and number of parameters for variational autoencoder.

**Table 1.1** Autoencoder layers and dimensions

Encoder		Decoder	
Layer Type	Output Shape	Layer Type	Output Shape
Input	28x28x1	Input	2
Reshape	28x28	Dense	16
LSTM	256	Dense	64
Dense	10	Dense	128
Dense	10	Reshape	4x4x8
Lambda	10	Convolution Transpose	8x8x8
		Convolution Transpose	12x12x4
		Convolution Transpose	24x24x2
		Convolution Transpose	28x28x1

**Table 1.2** GAN layers and dimensions

Discriminator		Generator	
Layer Type	Output Shape	Layer Type	Output Shape
Convolution	14x14x64	Dense	6272
Leaky Relu	14x14x64	Leaky Relu	6272
Dropout	14x14x64	Reshape	7x7x128
Convolution	7x7x64	Convolution Transpose	14x14x128
Leaky Relu	7x7x64	Leaky Relu	14x14x128
Dropout	7x7x64	Convolution Transpose	28x28x128
Flatten	3136	Leaky Relu	28x28x128
Dense	1	Convolution	28x28x1

## 1.2. Handwritten Digit Generation with VAE

Train your model and plot the values of the loss function and the KL divergence term during training. Comment on their behaviors.

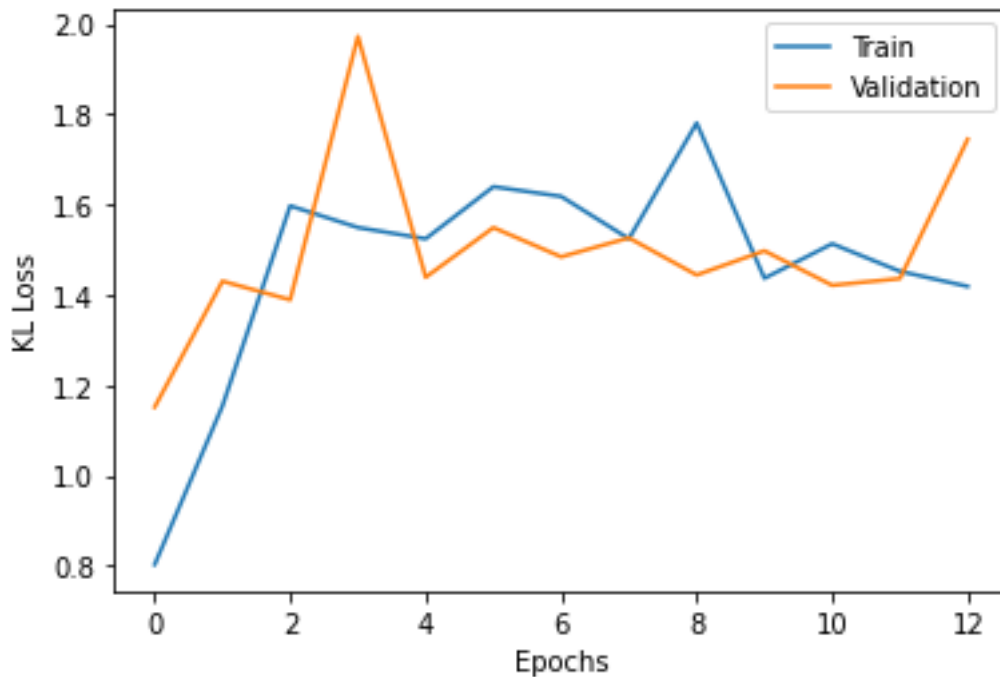


Figure 1.1 KL Loss for train and validation during epochs

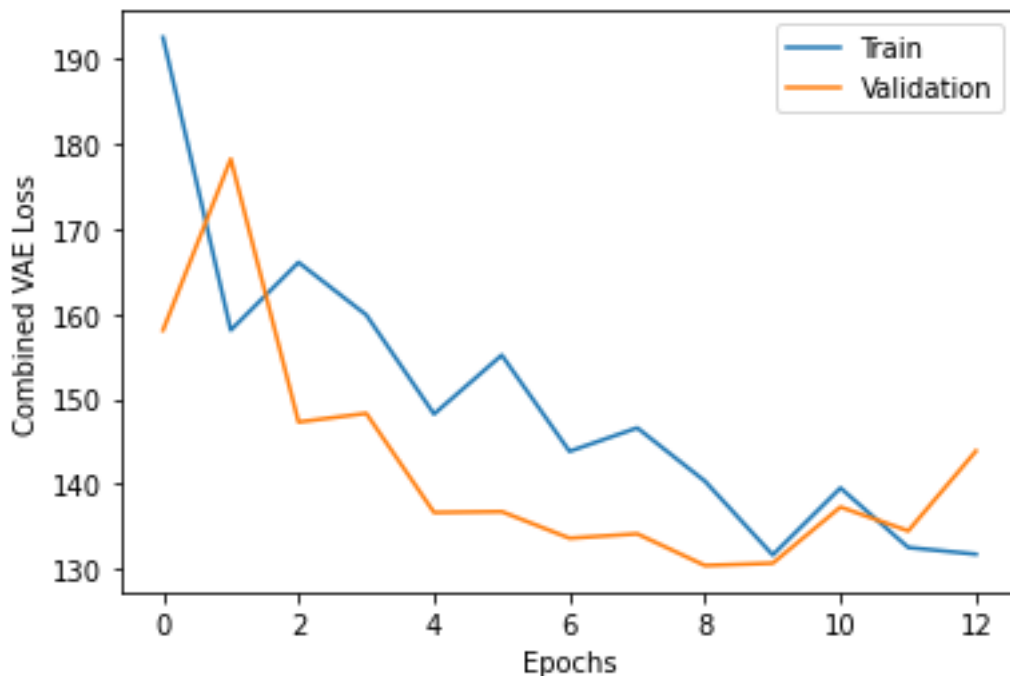
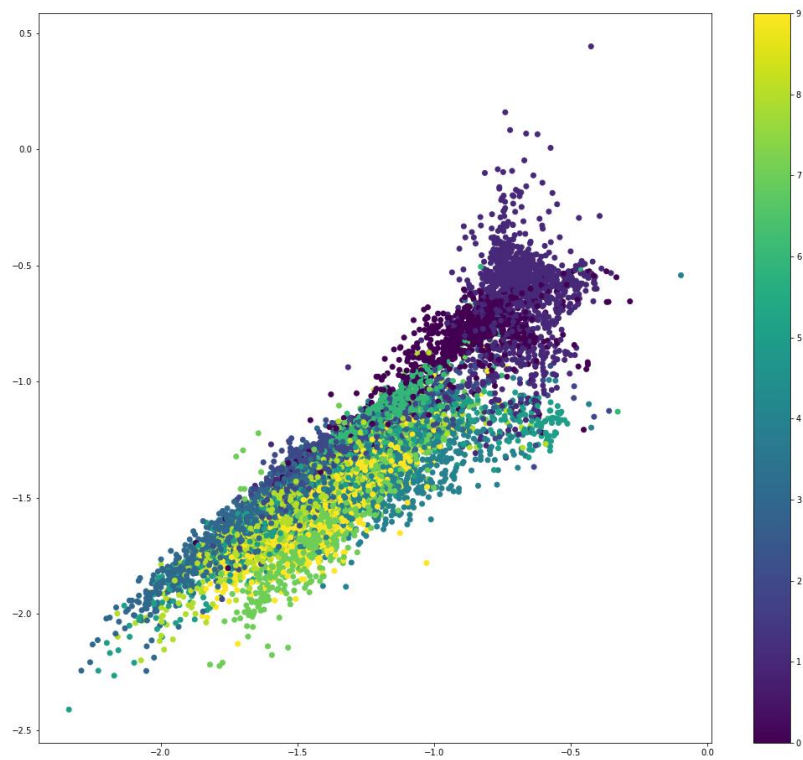


Figure 1.2 Combined Loss for train and validation during epochs

In variational autoencoders, reconstruction loss ensures how close the generated image is close to original images. On the other hand, KL divergence ensures how much two distribution is far from each other. In my experiments, generated images are not too bad. However, when I checked the distributions, different labels generally were nested together. It can be observed from following

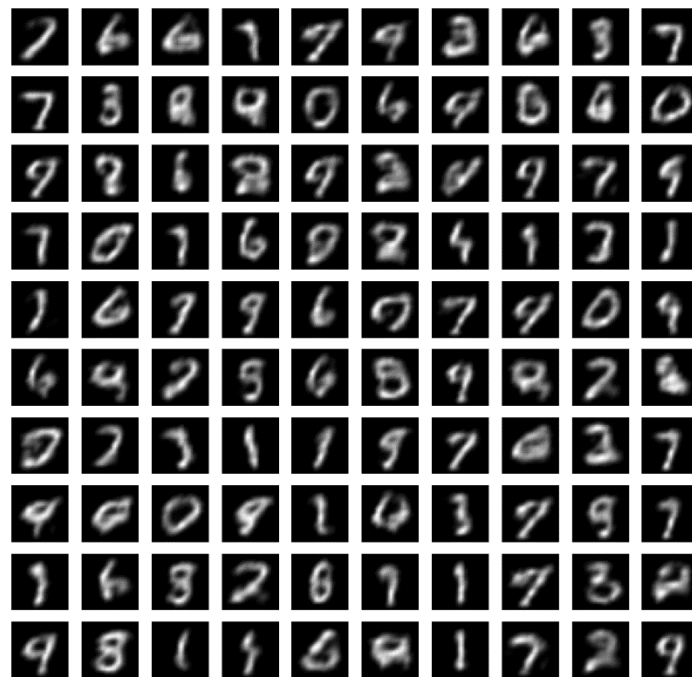
figure. This distribution was plotted for 2-dimensional latent vector experiment to understand result distribution. Also, I want to highlight that during training validation loss for combined loss is monitored. Early stopping is applied with patience of 4. The model in the epoch where the least loss is achieved is saved. In Figure 1.2, it can be observed that this is epoch 8.



**Figure 1.3** VAE generated image distributions

Increasing loss for KL divergence and Figure 1.3 shows us, I failed to train good encoder. The encoder is not capable of creating distinct distributions.

**Load the trained decoder and generate 100 images from 100 random vectors. Visualize the generated samples. Comment on your performance.**



**Figure 1.4** 100 generated image by VAE

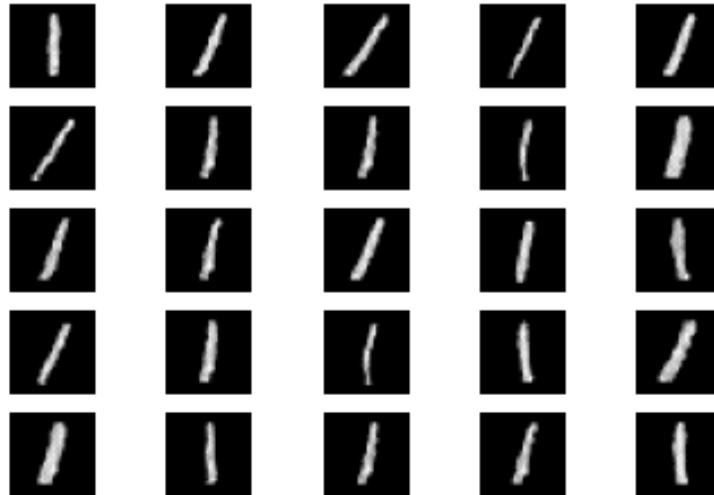
The created images by VAE are very noisy. It is common problem to produce noisy images with VAE. The reason behind that might be decreasing information to small bottleneck latent vector. During this process, it is possible to lose some information.

The numbers generally look like handwritten numbers with low quality. There are not random shapes. With deeper VAE, it might be possible to generate more realistic images. Since distributions are inside each other, some numbers look like combination of two numbers. However, training more epochs is not contributed too much after a while. Therefore, I think problem might be the architecture.

### **1.3 Handwritten Digit Generation with GAN**

**Comment on the relationship between the complexity of the generator and the generation performance.**

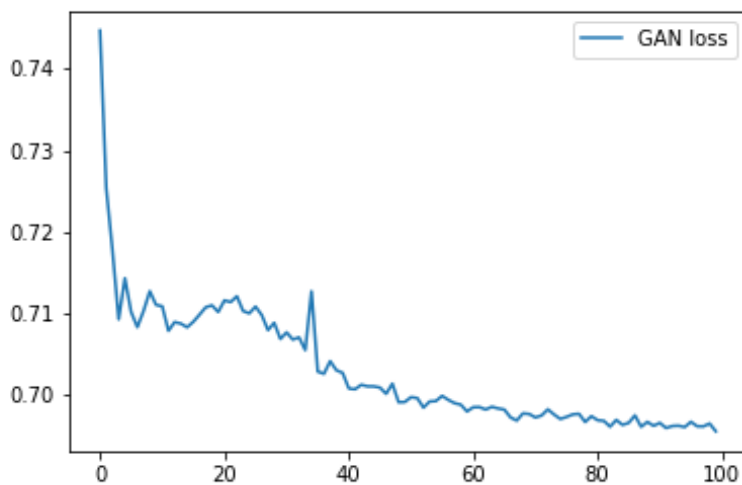
The first model I designed was unsuccessful to produce different images. After I searched about it, I learned that it happens because the generator was too simple compared to discriminator architecture. In other words, generator was not able to produce images that fool discriminator. Therefore, it sticks to one type of image that can fool discriminator and continuously learn to produce that. This problem is named in the literature as mode collapse. Following image is the examples of produced images by that generator.



**Figure 1.5** Mode collapse

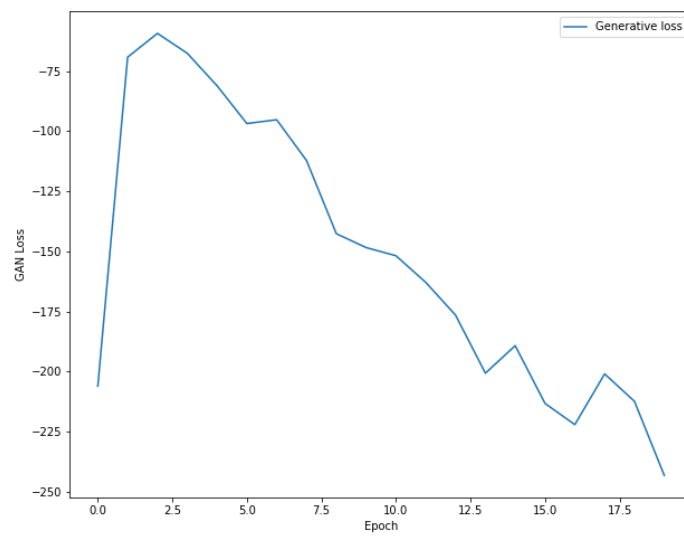
As it can be seen, the generator is always producing image of one. To achieve this problem, I designed more complex generator. Then, generation performance is increased considerably.

**Please train your GAN with cross-entropy loss and Wasserstein loss. Compare their convergence and stability.**



**Figure 1.6** GAN Loss with cross entropy loss function

For cross entropy loss function there is quite stable decreasing view. It converges between 80-100 epochs.



**Figure 1.7** GAN Loss with Wasserstein loss function

In Figure 1.7, the same model is trained with Wasserstein loss function. It can be observed that it is less stable and not converged yet.

Visualize the generated samples. (GAN)

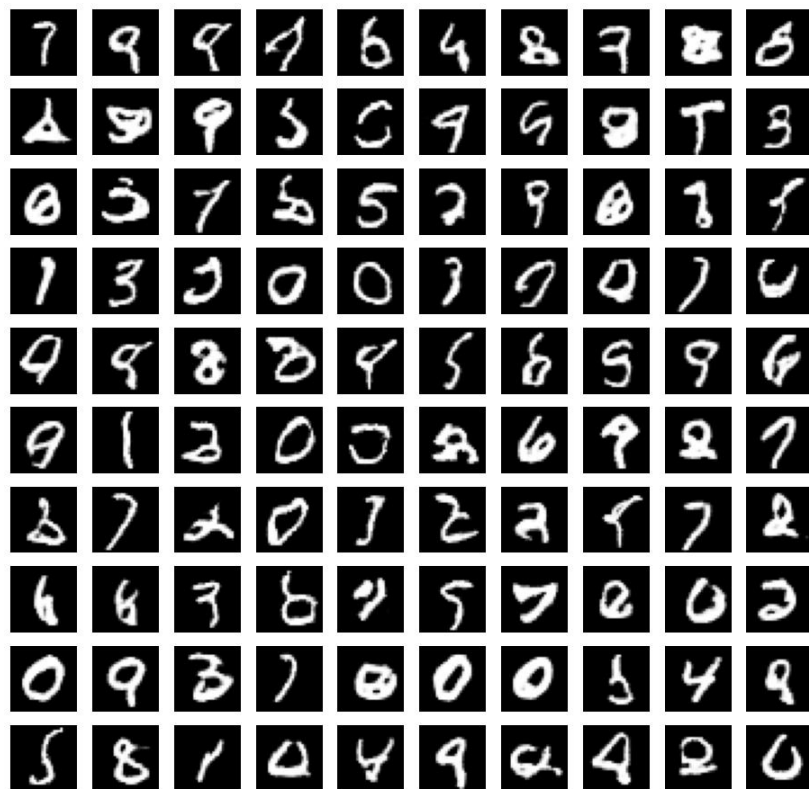


Figure 1.8 100 generated images by GAN

Compare the GAN and VAE performances both qualitatively and quantitatively.

It is hard to say something about success of the GANs since there is no objective measure of model performance. As it is done in this homework, generally model is saved for systematically during epochs. However, there are methods to evaluate generated images both qualitatively and quantitatively.

### Qualitative Analysis







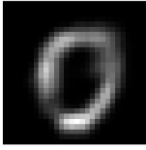

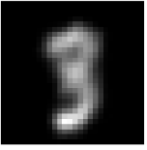







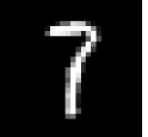

Normal MNIST						
VAE Generated						
GAN Generated						

Figure 1.9 Qualitative comparison

In Figure 1.8, I tried to put comparable examples in order from original dataset MNIST, images generated by VAE and GAN, respectively. First, in Figure 1.7, it can be observed that there are some



random shapes that are like handwritten numbers. This did not happen for the VAE generated images. It might be occurred because of overfitting. On the other hand, in Figure 1.8, it is clear that images generated by GAN are more realistic than VAE generated.

### Quantitative Analysis

For quantitative analysis, I wanted to apply inception score (IS) since it is the most applied metric. However, one of the most common mistakes for this metric is applying objects that are not available in ILSVRC 2012 dataset. There are 1000 different images in this dataset but MNIST labels are not available. Therefore, I modified the strategy for MNIST dataset. Instead of InceptionV3 model, I used pre-trained model on MNIST dataset. In inception case, the maximum score that can be achieved is 1000. For MNIST case, this decreases to 10. As a benchmark point, I first calculated inception score for original training MNIST dataset where 60000 images are available. The result was 9.50. Then, I calculated score for 500 images generated by both GAN and VAE. GAN is 7.52 and VAE is 1.72. This shows us GAN is more successful to generate images compared to VAE.

**Table 1.2** Inception score for MNIST dataset

	Inception Score (Average)	Standard Deviation
<b>Benchmark (Normal MNIST)</b>	9.50	0.01
<b>VAE</b>	1.72	0.25
<b>GAN</b>	7.52	0.58

## 2. REFERENCES

- [1] Brownlee J. How to Develop a GAN for Generating MNIST Handwritten Digits. Machine Learning Mastery. 2019. <https://machinelearningmastery.com/how-to-develop-a-generative-adversarial-network-for-an-mnist-handwritten-digits-from-scratch-in-keras/>
- [2] Brownlee J. How to Evaluate Generative Adversarial Networks. Machine Learning Mastery. 2019. <https://machinelearningmastery.com/how-to-evaluate-generative-adversarial-networks/>
- [3] Brownlee J. How to Implement the Inception Score (IS) for Evaluating GANs. Machine Learning Mastery. 2019. <https://machinelearningmastery.com/how-to-implement-the-inception-score-from-scratch-for-evaluating-generated-images/>
- [4] Pawangfg. Role of KL-divergence in Variational Autoencoders. Geekforgeeks. 2022. <https://www.geeksforgeeks.org/role-of-kl-divergence-in-variational-autoencoders/>
- [5] Shafkat I. Intuitively Understanding Variational Autoencoders. TowardsDataScience. 2018. <https://towardsdatascience.com/intuitively-understanding-variational-autoencoders-1bfe67eb5daf>
- [6] Borji, A. (2018). Pros and Cons of GAN Evaluation Measures. arXiv:1802.03446.
- [7] Brownlee J. How to Develop a Wasserstein Generative Adversarial Network (WGAN) From Scratch. Machine Learning Mastery. 2019. <https://machinelearningmastery.com/how-to-code-a-wasserstein-generative-adversarial-network-wgan-from-scratch/>
- [8] ray0809. 2018. Simple WGAN with MNIST. <https://github.com/ray0809/simple-wgan-with-mnist>
- [9] Anwar A. Difference between AutoEncoder (AE) and Variational AutoEncoder (VAE). 2021. TowardsDataScience. <https://towardsdatascience.com/difference-between-autoencoder-ae-and-variational-autoencoder-vae-ed7be1c038f2>
- [10] DigitalSreeni. 126 - Generative Adversarial Networks (GAN) using keras in python. 2020. Youtube. [https://www.youtube.com/watch?v=Mng57Tj18pc&t=468s&ab\\_channel=DigitalSreeni](https://www.youtube.com/watch?v=Mng57Tj18pc&t=468s&ab_channel=DigitalSreeni)
- [11] DigitalSreeni. 130 - Evaluating the deep learning trained model (Keras and TensorFlow). 2020. Youtube. [https://www.youtube.com/watch?v=MSjW3qyw7H0&ab\\_channel=DigitalSreeni](https://www.youtube.com/watch?v=MSjW3qyw7H0&ab_channel=DigitalSreeni)
- [12] DigitalSreeni. 129 - What are Callbacks, Checkpoints and Early Stopping in deep learning. 2020. Youtube. [https://www.youtube.com/watch?v=wkwtIeq9Ljo&ab\\_channel=DigitalSreeni](https://www.youtube.com/watch?v=wkwtIeq9Ljo&ab_channel=DigitalSreeni)
- [13] DigitalSreeni. 131 - How to load a partially trained deep learning model and continue training?. 2020. Youtube. [https://www.youtube.com/watch?v=4umFSRPx-94&ab\\_channel=DigitalSreeni](https://www.youtube.com/watch?v=4umFSRPx-94&ab_channel=DigitalSreeni)
- [14] DigitalSreeni. 131 - How to load a partially trained deep learning model and continue training?. 2020. Youtube.
- [15] DigitalSreeni. [https://github.com/bnsreenu/python\\_for\\_microscopists](https://github.com/bnsreenu/python_for_microscopists)
- [16] Chollet F. Building Autoencoders in Keras. Keras. 2016. <https://blog.keras.io/building-autoencoders-in-keras.html>