NN for Images – Ex3

David Guedalia

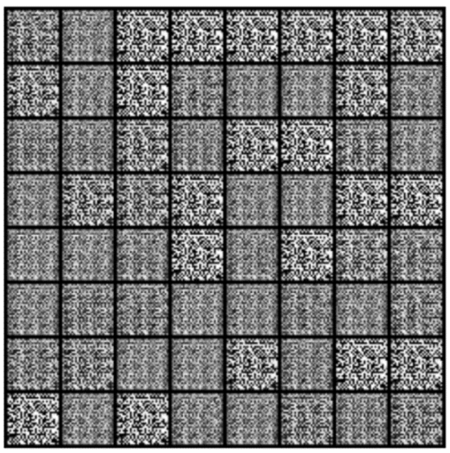
1. Practical Task

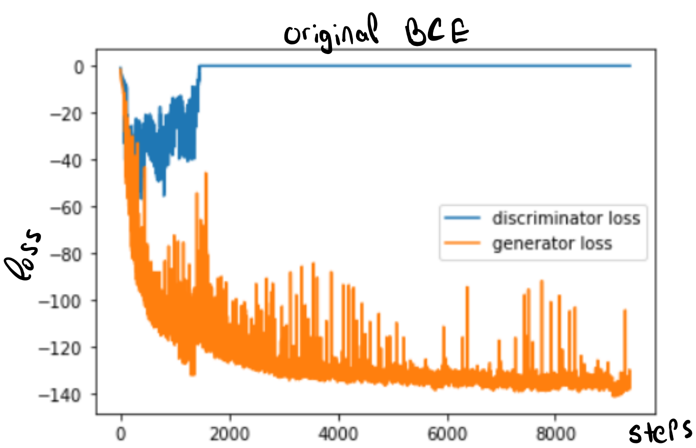
GAN

1.1 Loss Saturation

- On this section I followed the DCGAN architecture, I made a few changes of this architecture that I found improve the performances.
- I trained 3 steps on the discriminator per one single iteration of the generator.
- I implemented the generator with layers of conveTranspose, ReLU and batchNorm where I used Sigmoid ad the last layer.
- I implemented the discriminator with layers of conve, LeakyReLU and batchNorm where I used Sigmoid ad the last layer, I used LeakyReLU as I saw that it make the generator better results.

1.1.1 The original GAN cross entropy

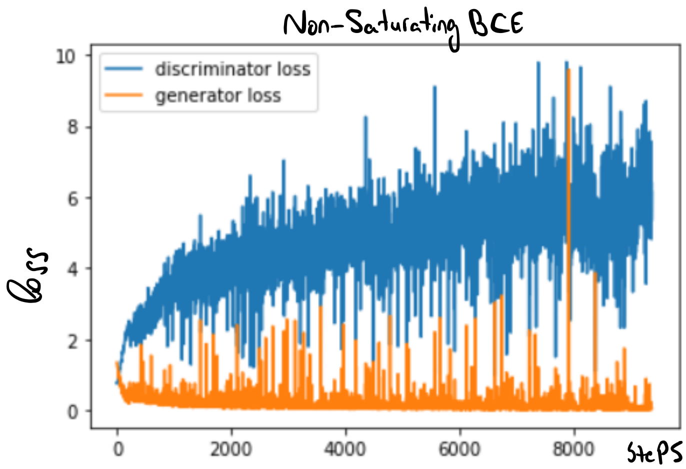




The results were bad, and as can see it didn’t learn, I can explain it as the discriminator converge much faster than the generator, and the discriminator have very small gradients and that’s why he can't learn well.

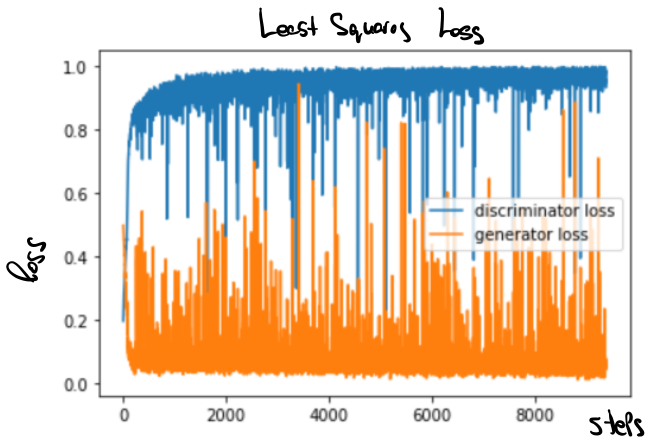
1.1.2 The Non-Saturating BCE:

We can see much better results, and even the best results from the three options, I think its because the loss maximize the likelihood of the discriminator that will be wrong –(for training the generator).



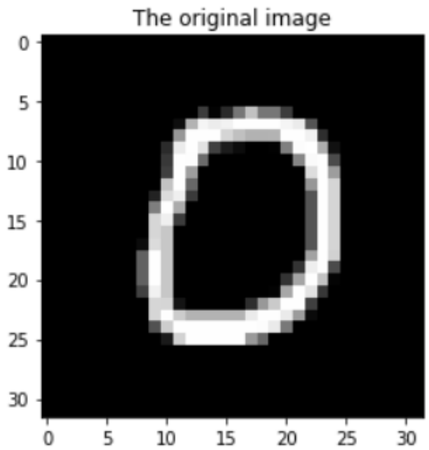
1.1.2 The Least Squares Loss

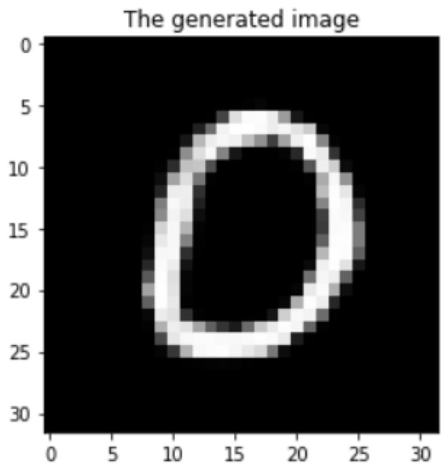
This loss also achives very good results, although it is not better then the non-saturating but we can see a very good results.

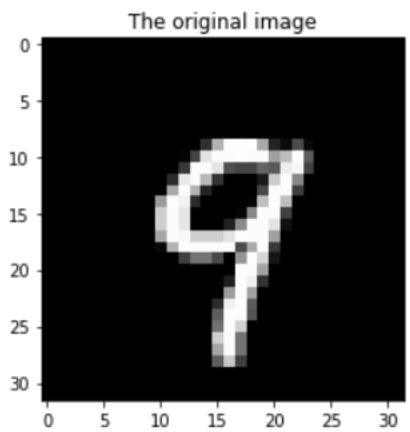


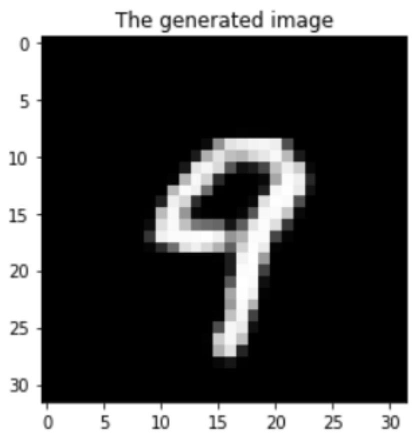
1.2 Model Inversion

In this section for given a picture I optimize over z and I used the MSE, means to find for image the vector z that the generator make the image.
I used images from the test, as we can see the inverse image looks like the original image, we can explain it as the generator were trained on similar images, also I found that the typical images gives better performance.









Non-Adversarial Generative Networks

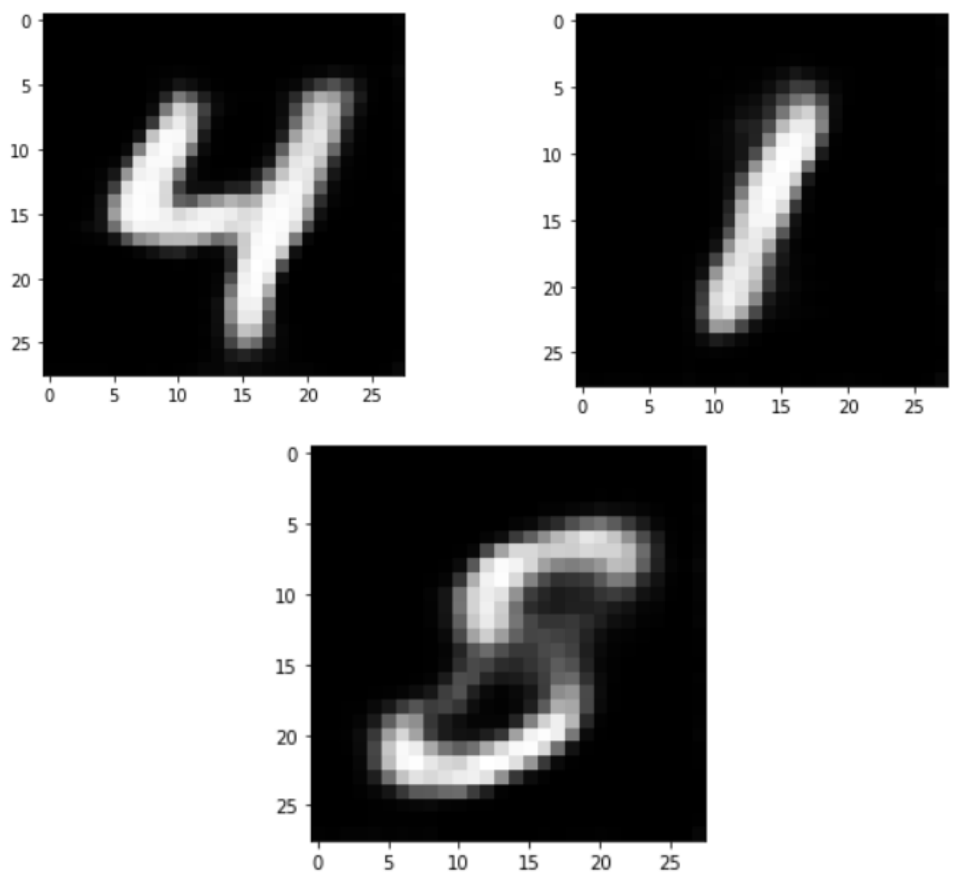
2.1 Wasserstein Auto-Encoders

In this section I reused code from the last exercise, I changed the loss, and add to the loss that latent space from the encoder to be close to the standard gaussian distribution.

Basically, I used two functions of loss one for the decoder and the other for the encoder, and loss it between the gaussian of the encoder and the standard gaussian distribution.

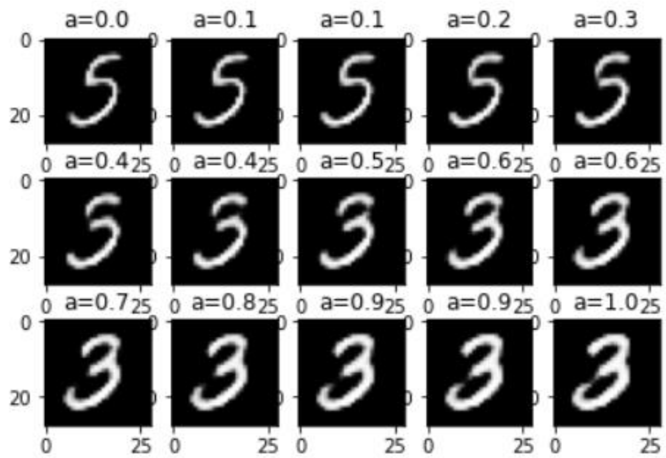
From the results we can conclude that the decoder does not have good performance on the standard gaussian distribution comparing to the generator.

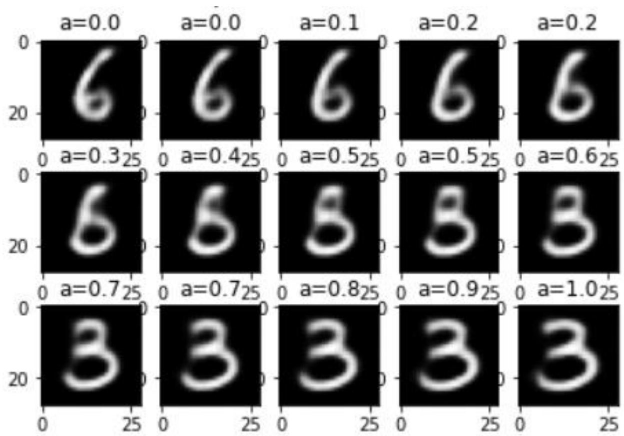
I can explain it because the generator is trained with normal distribution, and the autoencoder is not trained on random normal distribution



2.2 Interpolation

As we can see the Wasserstein autoencoder gives more elegant transitions, although the normal AE from the last exercise gives better digits and more clearly.





תרגיל 3 - תיאורטי

שאלה 1

במודל Glow אנו משתמשים ברשת שניתן לכתוב אותה כהרכבה של פונק':

$$M = h_K \circ h_{K-1} \circ \dots \circ h_1$$

מאחר ובמודל זה אנו משתמשים בשכבות הפיכות אז ניתן להגדיר:

$$M^{-1} = h_1^{-1} \circ \dots \circ h_K^{-1}$$

כמו כן אנו משתמשים בשכבות שניתן לחשב ביעילות את היעקוביאן מאחר וכפי שראינו בהרצאה קל לחשב דטרמיננטה של מטריצה משולשית, ולכן ניתן להסיק כי:

$$P(x) = \left|\frac{\partial M^{-1}(x)}{\partial x}\right| P_z\left(M^{-1}(x)\right)$$

נשים לב כי לא ניתן לחשב את $P(x)$ ב GAN ו GLO מאחר ואנו לא משתמשים רק בשכבות הפיכות.
בנוסף כפי שראינו לא בהכרח החישוב של היעקוביאן יהיה פשוט וקל לחישוב.

שאלה 2

תחילה נסמן את $F_D(G(z))$ שמציין את הערך שהתקבל רגע לפני שהדיסקרימינטור מפעיל את הסיגמואיד ולכן נסמן $D(G(z)) := S(F_D(G(z)))$

כאשר S היא פונק' הסיגמואיד

נזכר:

$$V(D,G) = \mathbb{E}\log(D(x)) + \mathbb{E}\log(1-D(G(z)))$$

נגזור לפי G:

$$\nabla_G V(D,G) = \underbrace{\nabla_G \mathbb{E}\log(D(x))}_{=0} + \nabla_G \mathbb{E}\log(1-D(G(z)))$$

$$= -\frac{1}{1-D(G(z))}\cdot \nabla_G D(G(z))$$

$$= -\frac{1}{1-D(G(z))}\cdot S'(F_D(G(z)))\cdot \nabla_G F_D(G(z))$$

$$= -\frac{1}{1-\underbrace{D(G(z))}_{\approx 0}}\cdot \frac{-e^{-\overbrace{F_D(G(z))}^{\approx -\infty}}}{(1+e^{-F_D(G(z))})^2}\cdot \nabla_G F_D(G(z))$$

$$\approx -1\cdot 0\cdot \nabla_G(F_D(G(z))) \approx 0$$

כלומר נקבל כי $\nabla_G V(D,G) \approx 0$, כלומר הגרדיאנט שלו הוא 0 ולכן לא מצליח ללמוד, וזה גם מסביר את בעיית הגרדיאנט בחלק הקודם.

שאלה 3

סעיף א:

נשים לב כי:

$$\min_{G} V(D,G) = \underbrace{\mathbb{E}_{x\sim p_d} \log D(x)}_{\text{קבוע}} + \mathbb{E}_{z\sim p_z(z)} \log(1-D(G(z)))$$

בצורה הזאת אפשר לראות כי בגלל הפונקציה נמירת את G שממנמם את הנראות, רק ש G מנסה להטעות את D.

סעיף ב

בהמשך לסעיף הקודם הבעיה בהבנית שנתנה G אופטימלי, נאמר למקסם את הנראות של D כדי לסווג דגימה אמיתית ראמית עבור ה G האופטימלי. לדעתי בעיה זו קשה יותר עבור קיומה מיבחן של רשת נוירונים כי הבדל בין הדגימה האמיתית למזויף יכול להיות קטן ויהיה קשה עבור רשת זו לסווג.

סעיף ג

נניח כי D נכשל בשיפורים כלומר הגיע לא מצליח להבדיל בין דגימה אמיתית למזויף כלומר הגנרטור לא יוכל להשתפר יותר בגרדיאנט הבאה וכך ישאר את הגנרטור במקום.