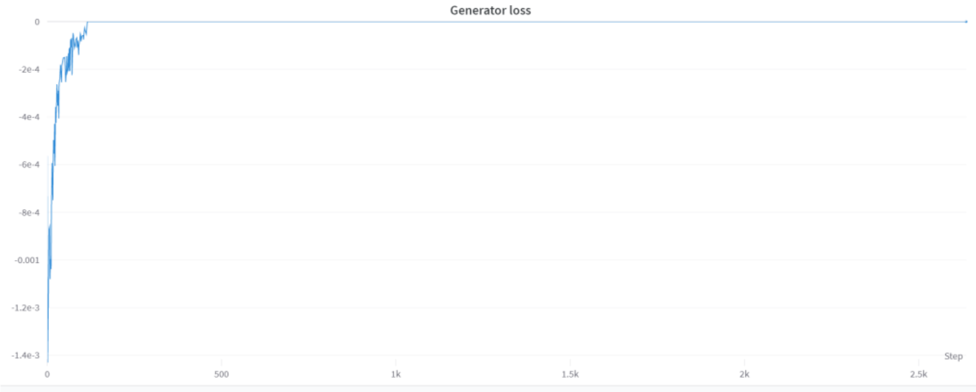
Practical part – Bar Rousso – I.D. 203765698

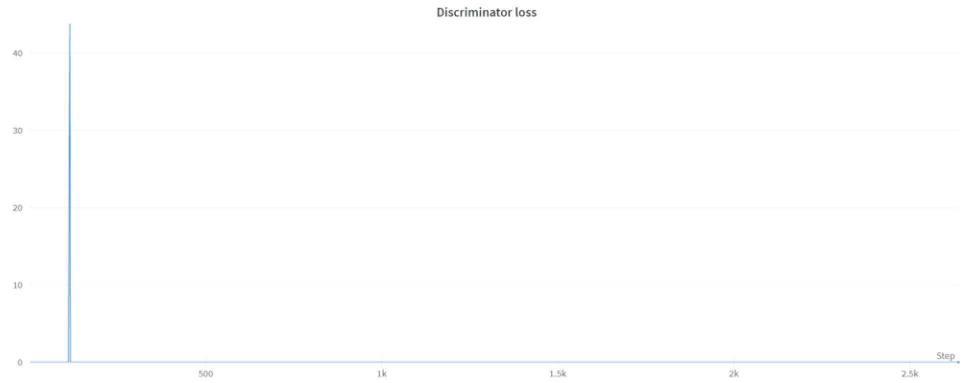
Question 1:

1. **Binary Cross Entropy Loss with saturation:**

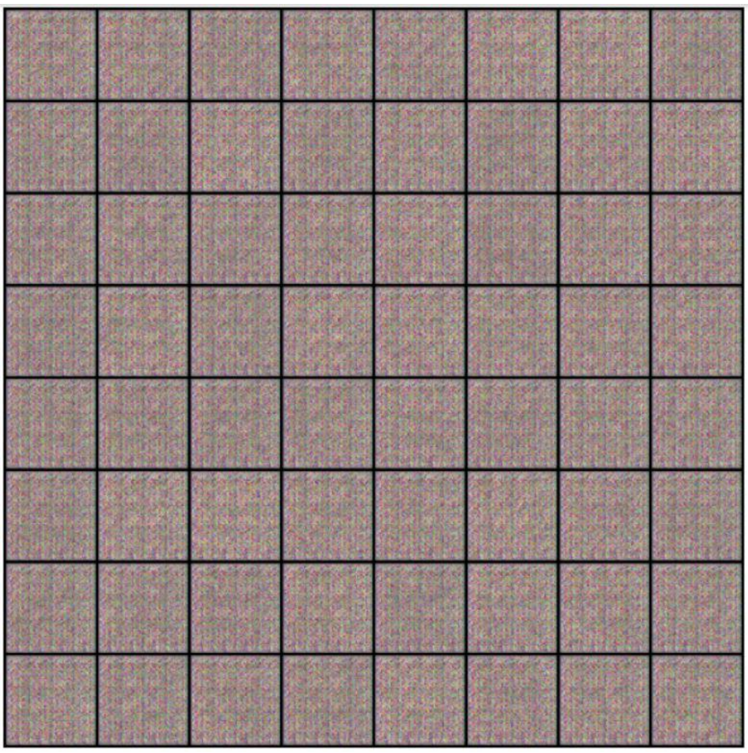
Discriminator loss function: $\mathrm{E}[\log(D(x))] + E[log\,(1 - D(G(z)))]$

Generator loss function: $E[lo\,g\left(1 - D\left(G(z)\right)\right)]$





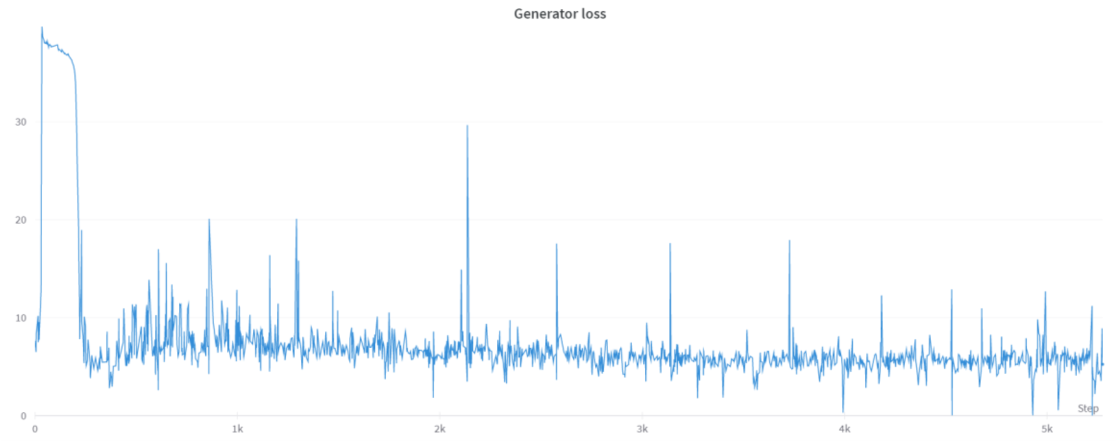
Final generated images from 64 fixed latent vectors:

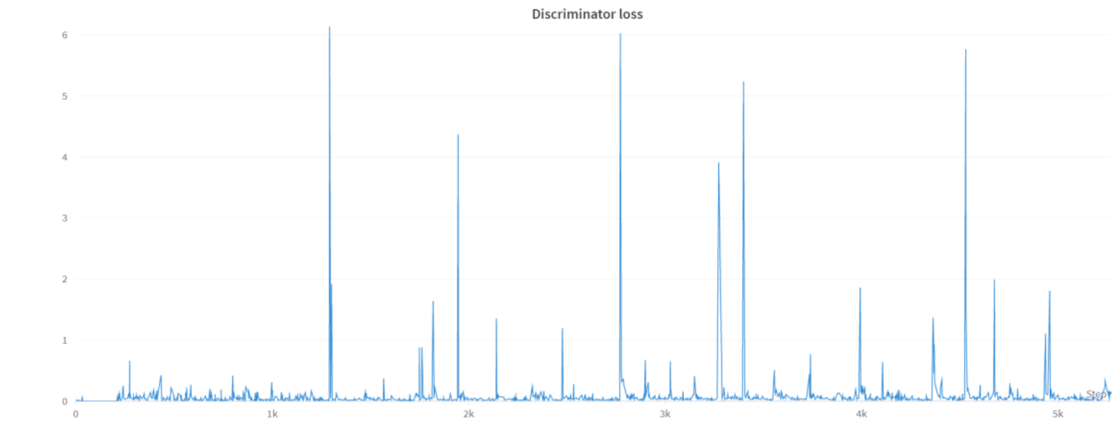


2. Binary Cross Entropy Loss without saturation:

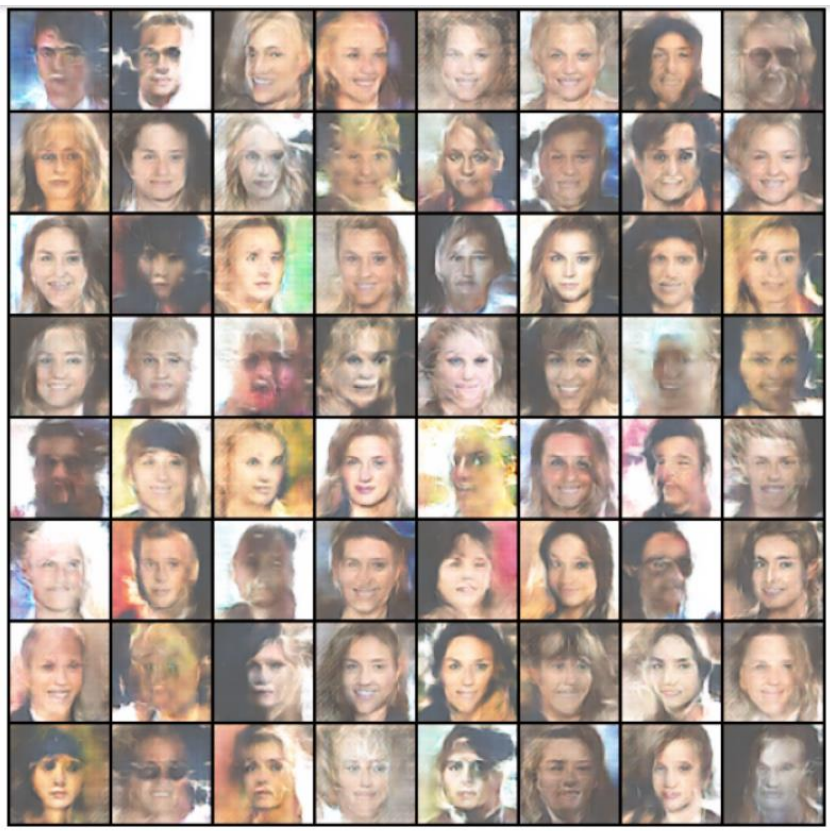
Discriminator loss function: $\mathrm{E}[\log(D(x))] + E[log\ (1 - D(G(z)))]$

Generator loss function: $E[-1 * lo\ g\left(D(G(z))\right)]$





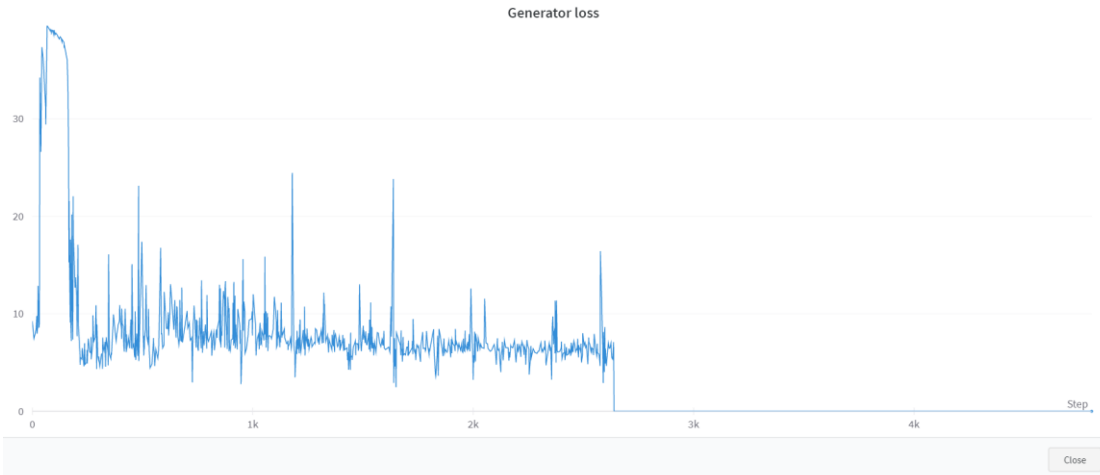
Final generated images from 64 fixed latent vectors:

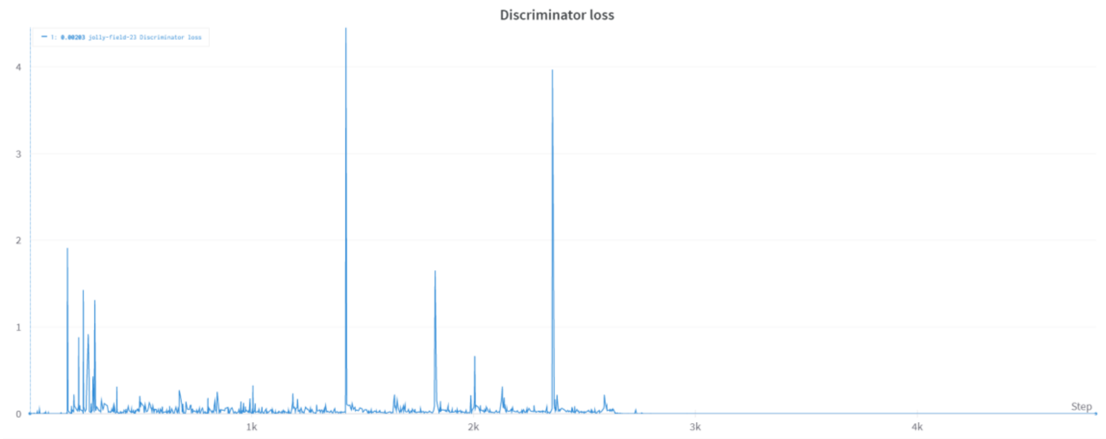


3. Mean Squared Error Loss

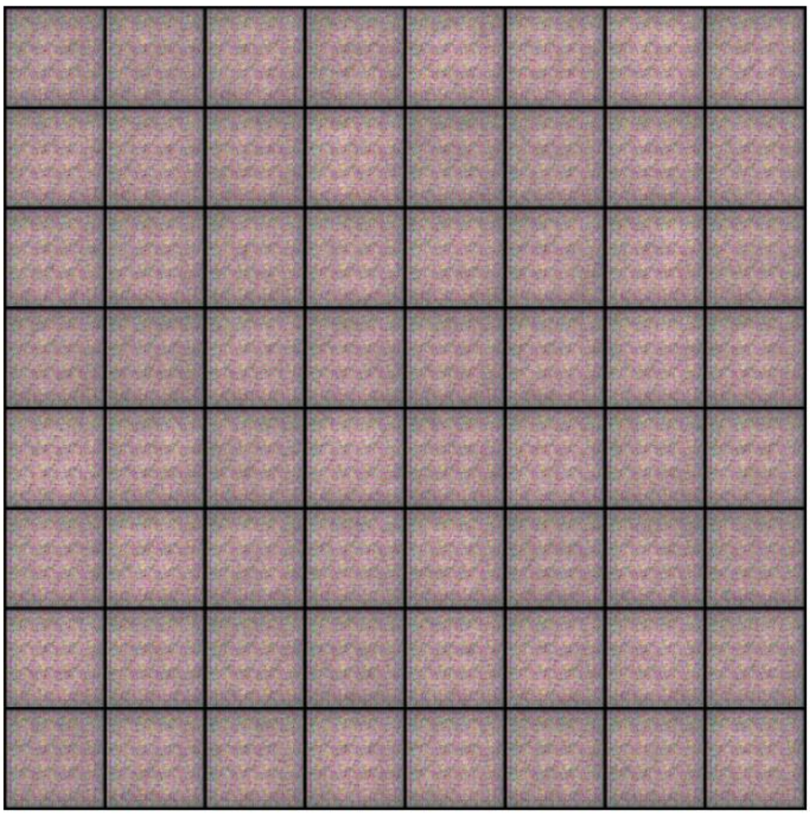
Discriminator loss function: $\mathrm{E}[\log(D(x))] + E[log\,(1 - D(G(z)))]$

Generator loss function: $(1 - D(G(z)))^2$





Final generated images from 64 fixed latent vectors:



Analysis:

In both the first and the last cases, we can see that we run into a saturation:
The **Discriminator** loss **tends to ZERO** - Meaning it succeeded to returns 1 on real images (hence $\mathrm{E}[\log(D(x))] = 0$) and returns 0 from generated images (hence $E[log\ (1 - D(G(z)))]$ = 0).

On the other hand, the **Generator** loss also **tends to ZERO** - Meaning it failed to convince the Discriminator that the images are generated is real.

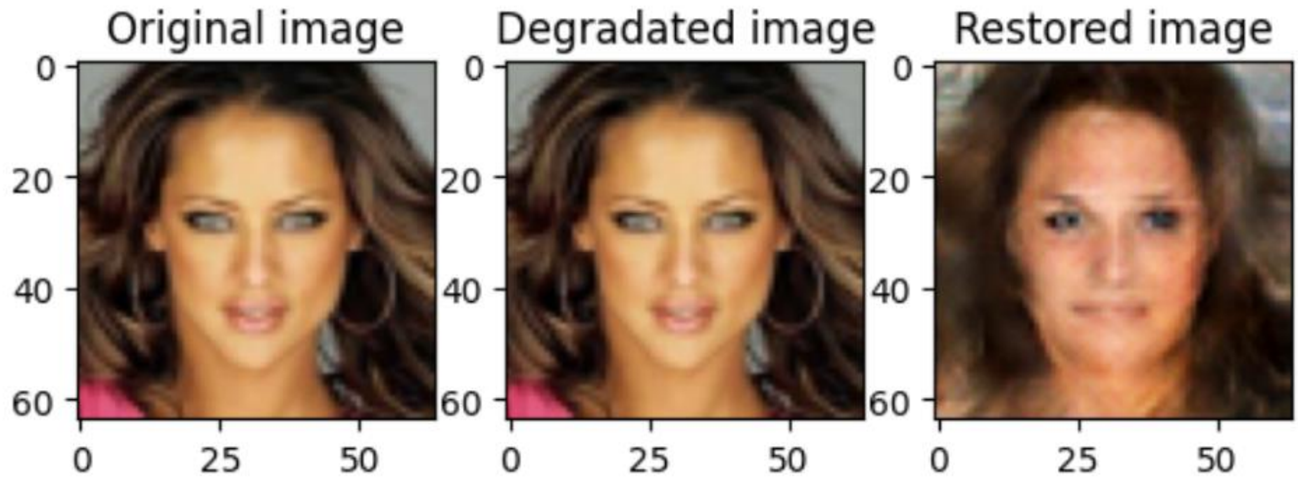
Thus, it is almost impossible for the generator to learn something about the real distribution, as the Discriminator is too strong – hence we get garbage final images.

In the second case, we can see that we did **not** run into a saturation:
The **Discriminator** loss tends to 0.05-0.1 (It is difficult to see from the graph since the high jumps) – thus compare to the first case the loss tends to 10^-8, here the discriminator is not equally decidable about the generator images.
On the other hand, the **Generator** loss tends to 5-7 (again it is difficult to see from the graph due to the high jumps) – Meaning it succeeded in some cases to make the discriminator think it made a real image.

Thus, the generator succeeded to learn something about the real distribution, as the Discriminator is not too strong – hence we get final images that remind us of people.

Question 2:

Final inversion after 400000 iterations:



Final MSE Loss: 0.007872

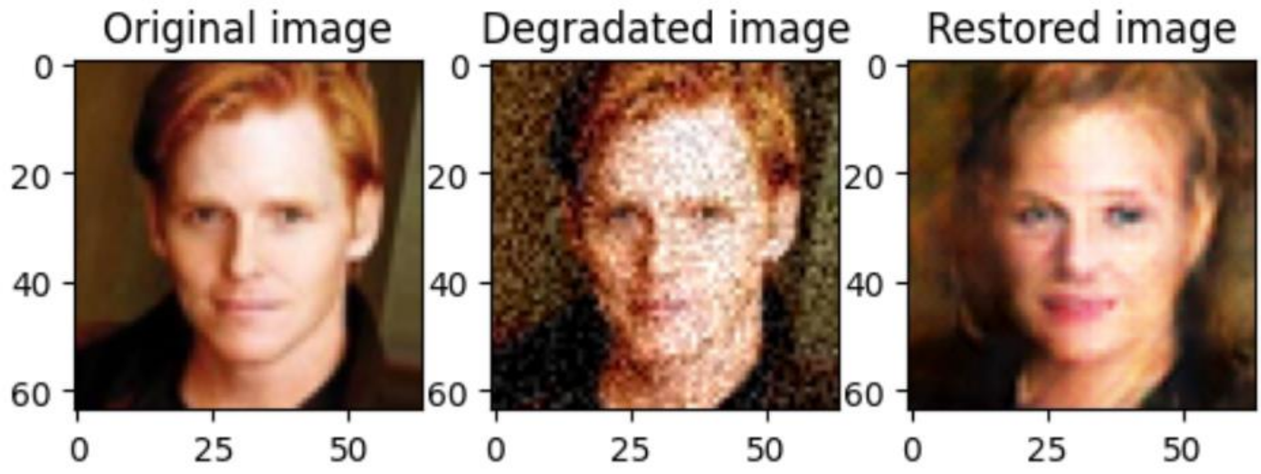
Note: There is **no** degradation here, we just want to find the closest image that our generator can produce.

Explanation:

We can see that we got an inverted image that "reminds" of the original image in terms of the colors (face, hair, background, etc.).
On the other hand, there the inverted image seems much more blur and miss thin details (hair edges, earring, face outlines, etc.). This can be explained since in our optimization progress, we use MSE Loss which is known to yield blur images.
Using perceptual loss can fix this issue by comparing both images' "features", rather than checking how much the pixel values are close to each other.

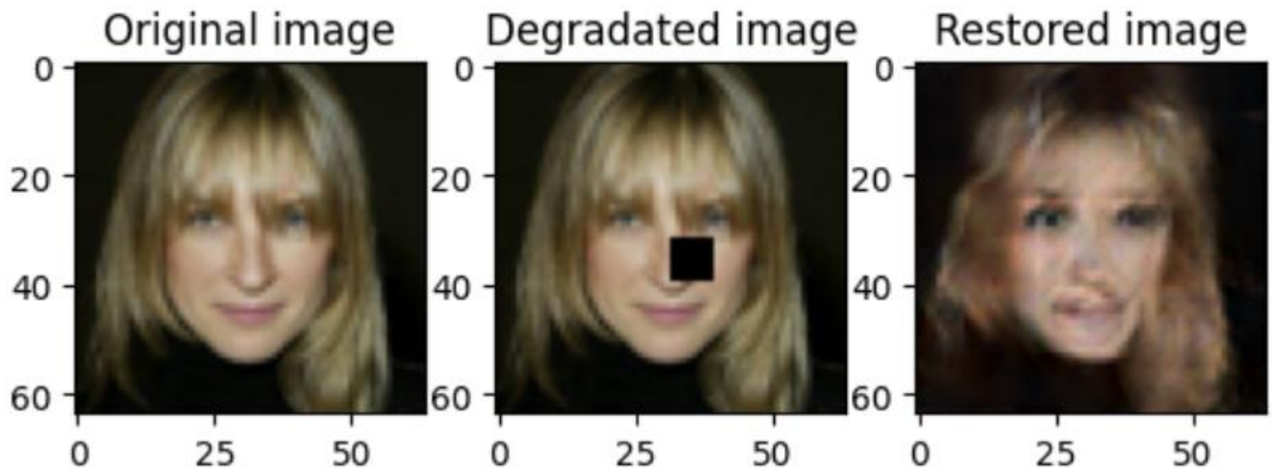
Question 3:

Section 1 – Restoring image after adding a gaussian noise:



Final L1 Loss: 0.0626

Section 2 – erase 8x8 window:



Final L1 Loss: 0.0512

Explanation (for both sections):

We can see that in both sections:
- We got restored images that remind their original images (color and shape are similar).
- The restored images indeed succeeded to remove the Gaussian noise or fill the missing window in a way that make sense (here restored to noise part).
- Still, both images are far from being a 'good' reconstruction – this mainly can due to the fact that the Generator didn't succeed to generate authentic images from the real distribution, as it is not complex enough, or can be trained with better hyper-parameters.

I choose to use the **L1 norm loss** since it is more invariant to the above degradations (compare to L2 norm) in a sense that differences in pixels caused due to degradation will be less expressed by the final Loss, since we are not raising this impact by 2.

חלק תיאורטי - משה בר חנן, ת.ז. 203465698

1) כאמור לפי מה שלמדנו בכיתה יש GLOW מורכב

מ 3 סוגי שכבות: עבור תמונת קלט X עם אורך h רוחב w

ומספר ערוצים C: לכל $1\le k\le C_{out}$, $1\le i\le h_{out}$, $1\le j\le W_{out}$:

$$(I)\quad y_{k,i,j}=[h_1(x)]_{k,i,j}=S\cdot x_{k,i,j}+b$$

כאשר S, b ערכים נלמדים והמימדים של הפלט לא משתנים

כלומר $W_{in}=W_{out}$, $h_{in}=h_{out}$, $C_{in}=C_{out}$

$$(II)\quad y_{k,i,j}=[h_2(x)]_{k,i,j}=W\bar{x}_{i,j}$$

כאשר $\bar{x}_{i,j}$ הוא וקטור הערוצים של הקלט X לכל $1\le i\le h_{in}$, $1\le j\le W_{in}$, ו- $W\in\mathbb{R}^{C_{out}\times C_{in}}$ זו מטריצה נלמדת.

ונשים לב כי ממדי השכבה והפלט נשמרים: $h_{in}=h_{out}$, $W_{in}=W_{out}$

$$(III)\quad y=h_3(x)=[y_a,y_b]$$

כאשר $i_1,..,i_a, j_1,...,j_b$ זו חלוקה כלשהי של שלשות האינדקסים (k,i,j) לכל $1\le k\le C_{in}$, $1\le i\le h_{in}$, $1\le j\le W_{in}$

ומתקיים כי $y_a=S\circ x_a+t$, $y_b=x_b$

כאשר $(\log S, t)=NN(x_b)$ עבור NN - רשת נוירונים כלשהי.

נניח כי רשת GLOW שלנו מורכבת מ n בלוקים של השכבות הנ"ל:

$$\underbrace{h_3\circ h_2\circ h_1}_{1}\circ\underbrace{h_3\circ h_1\circ h_1}_{2}\circ\dots\circ\underbrace{h_3\circ h_2\circ h_1}_{n}=M^{-1}$$

כמו כן, עבור $z\sim N(0,I)$, $x\sim P(x)$ נסמן ב- $P(x)$ את הצפיפות של תמונות שלמדנו.

ϵ

$$P(X) = e^{\log P(X)}$$

מ-קודם:

כאשר:

$$\log(P(X)) = \log(P_Z(M^{-1}(X))) + \log\left(\left|\det\left(\frac{\partial}{\partial X} M^{-1}(X)\right)\right|\right) =$$

$$= \log\left(P_Z\left(\prod_{t=1}^{n} [h_3 \circ h_2 \circ h_1](X)\right)\right) + \sum_{t=1}^{n} \sum_{q=1}^{3} \log\left(\left|\det\left(\frac{\partial h_q}{\partial h_{q-1}}\right)\right|\right)$$

הרכבה, ולא כפל!

$h_0 = h_3$

$$= \log\left(P_Z\left(\prod_{t=1}^{n} [h_3 \circ h_2 \circ h_1](X)\right)\right) + \sum_{t=1}^{n} \Big[h_{1,t_{in}} \cdot W_{1,t_{in}} \cdot C_{1,t_{in}} \cdot \log(|S_t|)\Big] +$$

$$+ \Big[h_{2,t_{in}} \cdot W_{2,t_{in}} \cdot \log(|\det(W_t)|)\Big] +$$

$$+ \Big[a_t \cdot \log(|\exp([NN(X_{b,t})]_0)|)\Big]$$

סימונים: לכל $1 \le t \le n$, $1 \le a \le 3$:

(I) $h_{a,t}$ = פונקציה h_a בבלוק t.

(II) $C_{a,t_{in}}$ = גודל מימד הערוצים בקלט של פונקציה $h_{a,t}$

(III) $W_{a,t_{in}}$ = אורך מימד הרוחב של הקלט של פונקציה $h_{a,t}$

(IV) $h_{a,t_{in}}$ = גודל מימד הגובה של הקלט של פונקציה $h_{a,t}$

(V) a_t = גודל התמונה $|\{i_1, \ldots, i_a\}|$ עבור פונקציה h_3 בבלוק t.

(VI) $X_{b,t}$ = וקטור שמכיל את האינדקסים $\{X_{j_1}, \ldots, X_{j_B}\}$ בחלוקה של פונקציה h_3 בבלוק t.

הסבר: נשים לב כי שיטות GAN, GRO שראינו ע"י generator המ"ג תמונות בהתפלגות דומה להתפלגות של ה DATA האמיתי. אך שיטות כגון GLOW, אין לנו צורך לחשב את ההתפלגות האמיתית, P(x), זאת מכיוון שהרשת של ה generator אין בהכרח מייצגת פונקציה הפיכה, כך נוכל שנקבל התפלגות עם חורים.

2) נזכר בבעיית ה min/max:

$$\min_G \max_D V(D,G) = \min_G \max_D E_{p_{data}}[\log(D(x))]+E_{p_z}[\log(1-D(G(z)))]$$

אם הדיון בבעיה הסטנדרטי, כלומר ש D כבר הינו δ classifier

חזק כך ש: עבור תמונה x מהדאטה $D(x)\approx 1$

עבור וקטור z $D(G(z))\approx 0$

הפונקציית Loss של G אותה נרצה למזער היא

$$L(G) = E_{p_{data}}[\log(D(x))]+E_{p_z}[\log(1-D(G(z)))]$$

עבור $t_G \leftarrow$ וקטור הפרמטרים כולל של G

נרצה לחשב את הגרדיאנט רק ב t_{G_0}.

$$\frac{\partial L(G)}{\partial t_{G_0}} = \frac{\partial}{\partial t_{G_0}}\left(E_{p_{data}}[\log(D(x))]+E_{p_z}[\log(1-D(G(t_G,z)))]\right)$$

$$= \frac{\partial}{\partial t_{G_0}}\left(E_{p_z}[\log(1-D(G(t_G,z)))]\right) = \frac{\partial}{\partial t_{G_0}}\left(E_{p_z}[f(t_G,z)]\right) = (*)$$

מכיוון שהנחנו, סטנדרטי ק"מ סביבה $B(t_{G_0},\epsilon)$ כך שלכל $t_G \in B(t_{G_0},\epsilon)$, עבור התפלגות אמפירי $D(G(t_G,z))$

קיים $\alpha<\delta_{t_G}$ כך ש $Pr[D(G(t_G,z))\geq\delta_{t_G}]=0$

לכן הגדרה עבור הנ"ל: $f(t_G,z):=\log(1-D(G(t_G,z)))$

$$Pr[|f(t_G,z)|\geq|\log(1-\delta_{t_G})|]=0$$

לכן נוכל כי $E_{p_z}[f(t_G,z)]<\infty$ לכן אפשר להחליף ∂ ו $\int$ ולקבל:

$$\frac{\partial L(G)}{\partial t_{G_0}} = (*) = E_{p_z}\left[\frac{\partial}{\partial t_{G_0}}(f(t_G,z))\right]$$

כעת נשים לב כי:

$$\frac{\partial}{\partial t_{G_0}} f(t_{G_0}, z) = \frac{\partial}{\partial t_{G_0}} \left(\log(1-x) \circ D(t_D, G(t_{G_0}, z)) \right) =$$

כאשר t_D הפרמטרים של D ו- z קבועים.

לפי הנחה

$$= \underbrace{\frac{1}{D(t_D, G(t_{G_0}, z)) - 1}}_{\mathrm{I}} \cdot \underbrace{\frac{\partial}{\partial G(t_{G_0}, z)} \circ D(t_D, G(t_{G_0}, z))}_{\mathrm{II}} \cdot \underbrace{\frac{\partial G}{\partial t_{G_0}} G(t_{G_0}, z)}_{\mathrm{III}}$$

אך נשים לב כי בהסתברות 1 כי

I - סופי, II - סופי היות ו D חסום, III - סופי

לכן $\frac{\partial}{\partial t_{G_0}} f(t_{G_0}, z) \approx 0$ ולכן: $\frac{\partial L}{\partial t_{G_0}} = E_{p_z}\left[\frac{\partial}{\partial t_{G_0}} f(t_{G_0}, z)\right] \approx 0$

כלומר, שהגרדיאנט בתהליך האימון של G מתאפס כמעט
לכן כמעט אין שינוי של הפרמטרים של G ולא "לומד"
אין למידה עבור G מכיוון ש D הגיע לסטורציה.

3) א) נסביר מה הבעיה $\min_G(V(D,G))$ מנסה לפתור:

במקרה הזה D קבוע !

בהינתן תמונה I שG מייצר $V(D,G)$ זה הציון
של D נותן לתמונה שאומר כמה הוא חושב כי I
זו תמונה אמיתית (מהתפלגות data) או מזויפת
(נוצרה ע״י G).

לכן הבעיה הנ״ל מנסה למצוא תמונה I
ספציפית (או z ספציפי כך ש $I=G(z)$)
כך ש $V(D,G)$ מינימלי ⇐ כלומר D חושב
שהיא הכי אמיתית (ציון הכי קרוב ל1).

(נשים לב שזו בדיוק הבעיה בmode-collapse)

ב) בהינתן שהבעיה $\min_G(V(D,G))$ התכנסה -
כאשר אומרת שG התכנס למשקולות ספציפיות כך
שלכל z $D(G(z))$ חושב בהסתברות קרובה
ל-1 שזו תמונה אמיתית / מהתפלגות data)

הבעיה $\max_D[\min_G(V(D,G))]$ מנסה למצוא

את המשקולות של D כך שינסה לתת ציון כמה שיותר
נמוך לתמונה שG מייצר וכמה שיותר גבוה לתמונה
האמיתית מה Data.

נשים לב שזו בעיה קשה לD, כאשר מניחים שG
כבר מייצר תמונות של D קשה להבחין בינן לבין התמונות
האמיתיות מה Data.

ג) דהיינו ש D לא מצליח להבדיל את הפתרון
אין תנאי שעוצר כדי להבדיל בין התמונה ש G
מייצר לתמונה האמיתית מה Data.
כך שאנו מקבלים אותנו נכנס למצב של mode-collapse
כלומר G ייצר רק חלק קטן מדוגמים מההתפלגות
האמיתית- Data שהינו דומים לייצר.