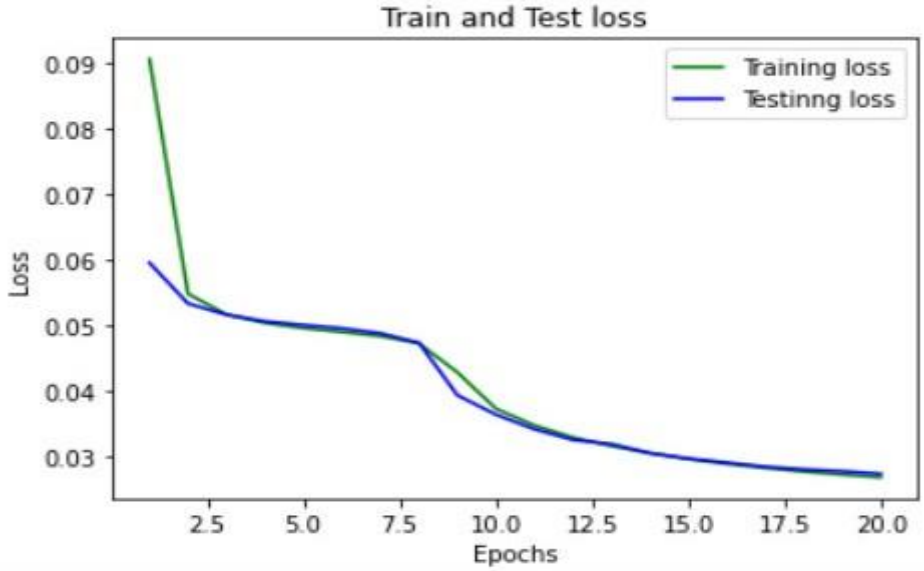
מבוא ללמידה עמוקה – תרגיל 3

מגישים:

שי כהן 314997388 איתי צ'אצ'י 208489732

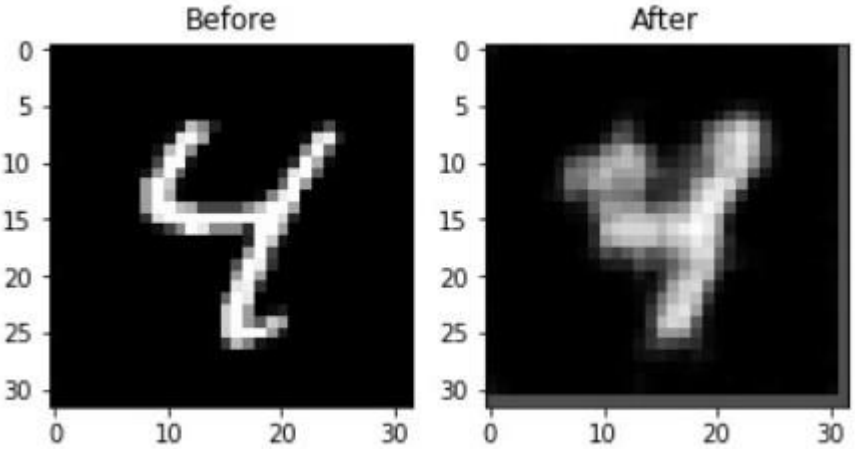
חלק מעשי:

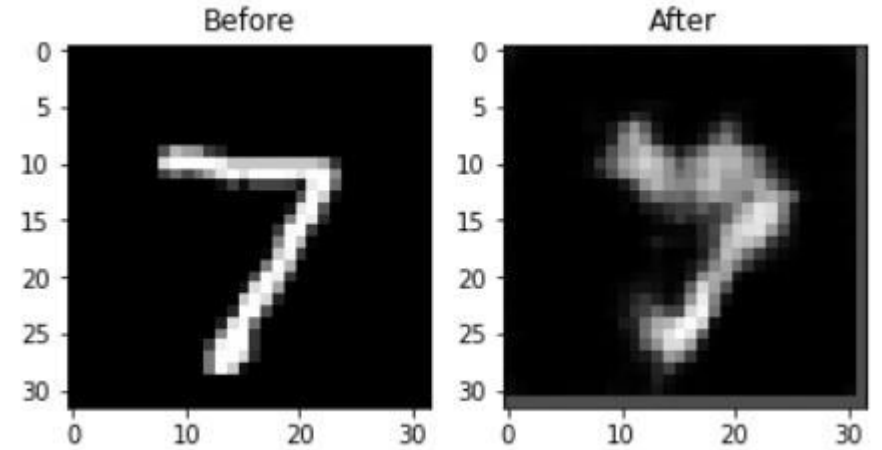
1) Autoencoders – בסעיף זה נדרשו לבנות AE בעל שכבות קונבולוציה שידע להוריד מימד לתמונה מ-MNIST וליצור אותה מחדש.
הגדרנו את גודל ה-latent space להיות 15 (נרחיב על בחירת גודל זה בסעיף הבא), חילקנו את הדאטה ל-60,000 דגימות אימון ו-10,000 דגימות טסט וביצענו 20 epochs.
מבחינת הארכיטקטורה של הרשת, השתמשנו בשכבות קונבולוציה ופונקציות ReLU ביניהן. בנוסף, הוספנו שכבות BatchNorm הן ב-Encoder והן ב-Decoder (ושכבות Flatten ו-Unflatten). תחילה, לא כללנו ברשתות שכבות FC.
נציג את תוצאות המודל ומספר דוגמאות לשחזור מספרים:

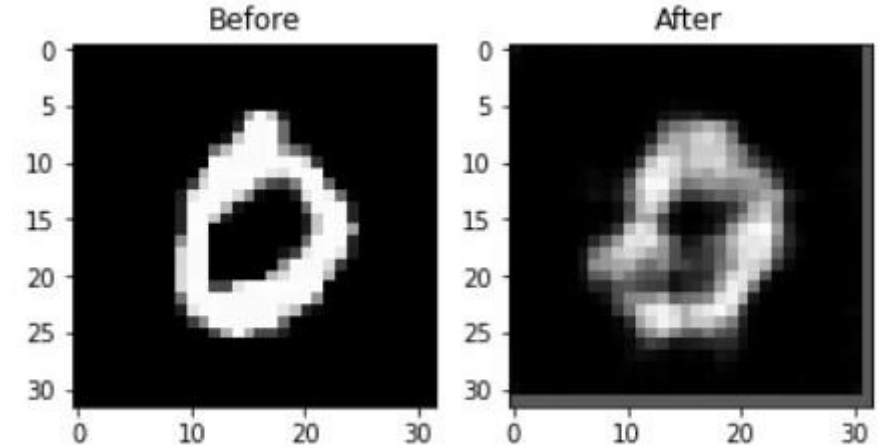


כפי שניתן לראות, אין overfit והמודל מתכנס ל-loss של בערך 0.03.

נציג מספר דוגמאות שחזור אותן הרצנו על המודל:

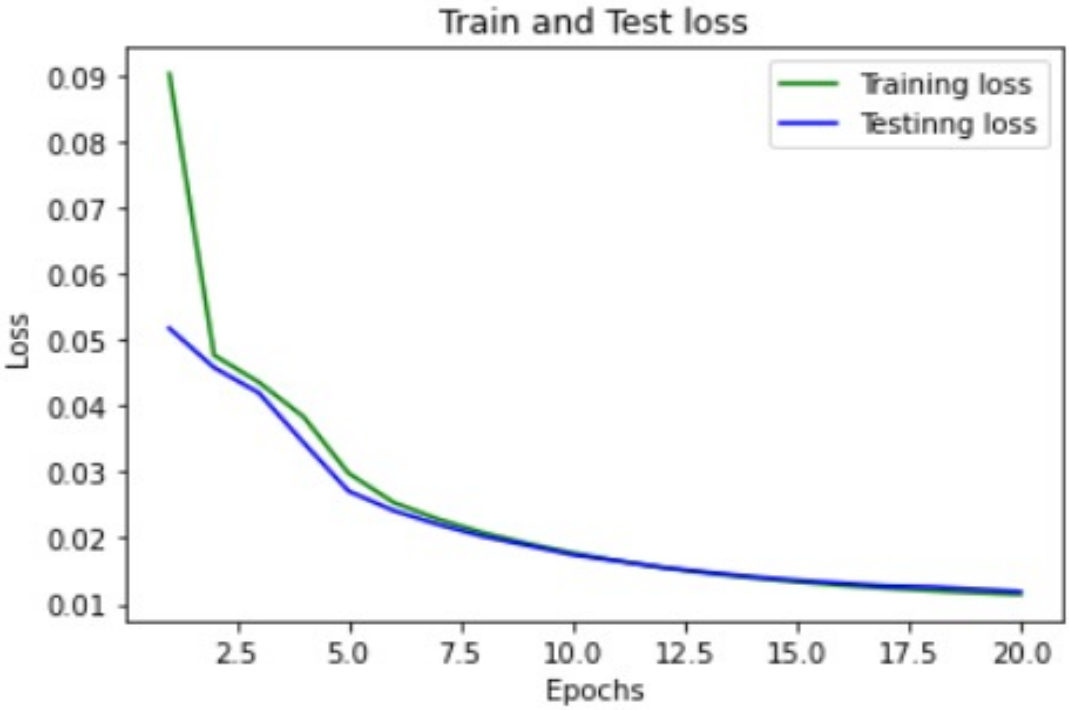


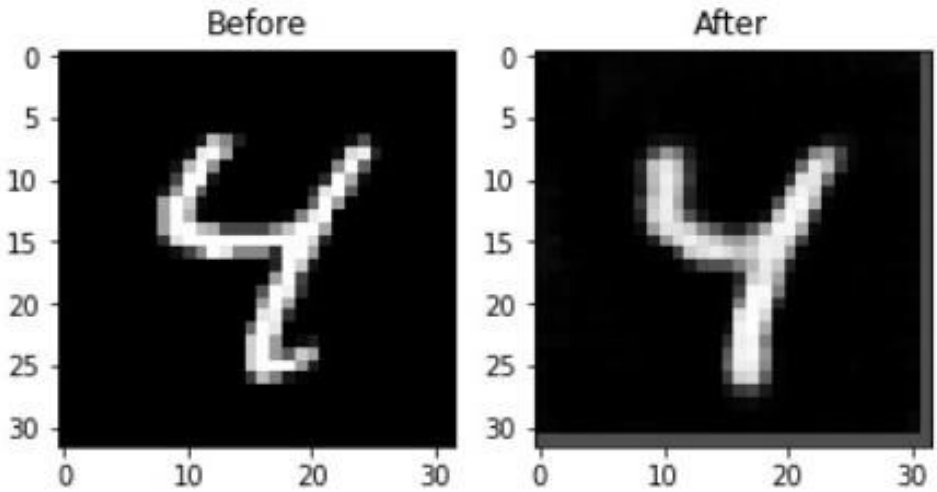


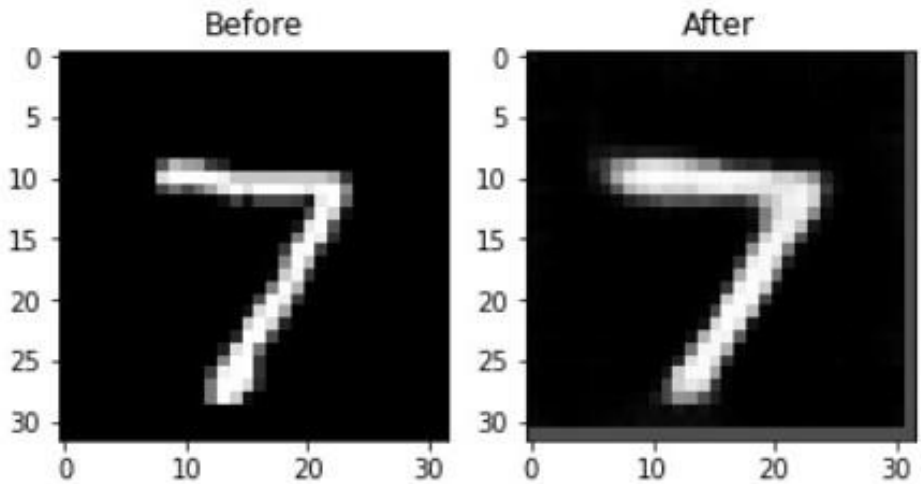


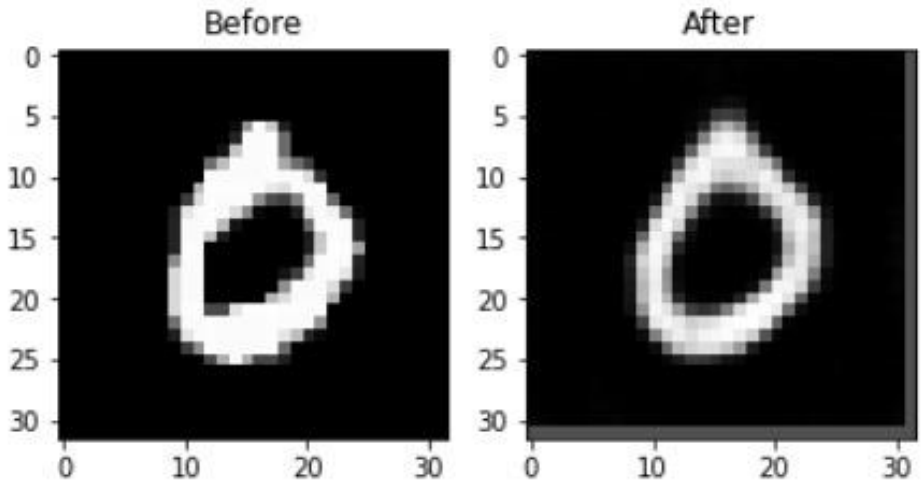
ניתן לראות, כי אכן המודל מתקרב לשחזור התמונה, וכי ניתן לזהות על סמך תמונת הפלט את המספר המקורי.
לאחר מכן, הוספנו שכבות FC ל-Encoder ול-Decoder לשכבות החיבור (הקרובות ל-latent vector). שמנו לב, כי תוספת זו מחזקת את ה-AE משמעותית ומפחיתה את ה-loss.

נציג את תוצאות המודל המשופר, ומספר דוגמאות לשחזור מספרים אשר ימחישו את חיזוקו:


Train and Test loss
Training loss
Testinng loss
Loss
Epochs
0.09
0.08
0.07
0.06
0.05
0.04
0.03
0.02
0.01
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0

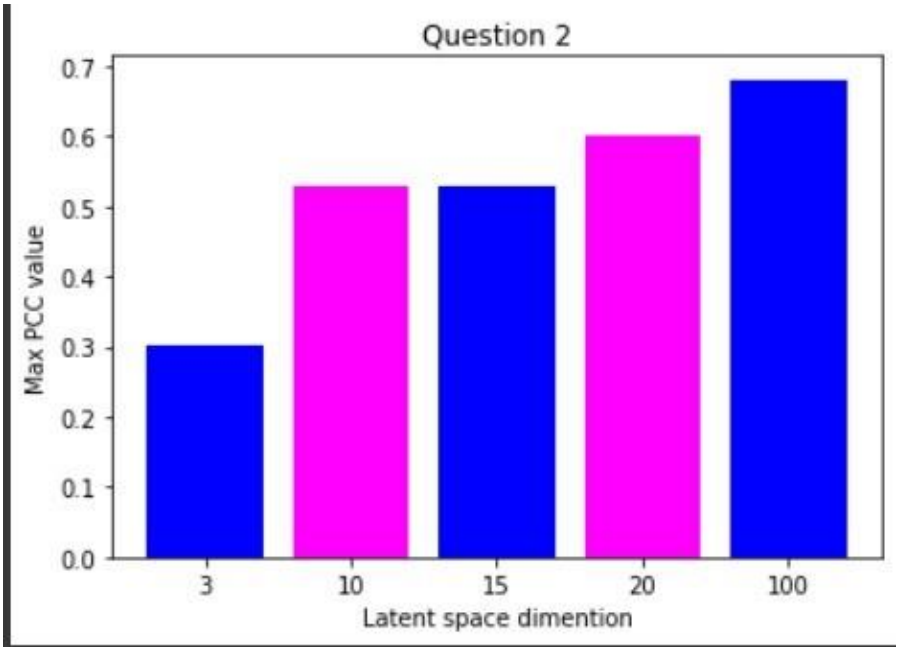

Before
After


Before
After



אכן ניתן לראות, כי המספרים ברורים בהרבה וכי תוצאות הרשת עם שכבות ה-FC משתפרות באופן משמעותי. ה-Loss מתייצב סביבות 0.01.

2) בשאלה זו, נדרשו לבחון את הקשר בין מימד ה-latent space למדד PCC של וקטור ה-latent. כדי לבחון קשר זה, אימנו את המודל 5 פעמים, עם גדלי latent space משתנים (3, 10, 15, 20, 100).
לאחר מכן, שמרנו את הוקטורים שה-Encoder פלט ומהם יצרנו מטריצת correlation ובחנו מספר דרכים כדי לבחון מדד זה על פי המטריצה.
הדרך הראשונה אותה ניסינו הייתה לחשב לכל מטריצה, את הממוצע של הערך המוחלט שלה. נכחנו לדעת, כי מדד זה אינו אינפורמטיבי ולא תורם להבנה.
לבסוף, בחרנו לקחת מכל מטריצה, את הערך המקסימלי שלה בערך מוחלט (בלי האחדות באלכסון הראשי) וכך בחרנו להעריך את הPCC שלה. אצרף גרף המציג מדד זה לכל גודל latent vector:

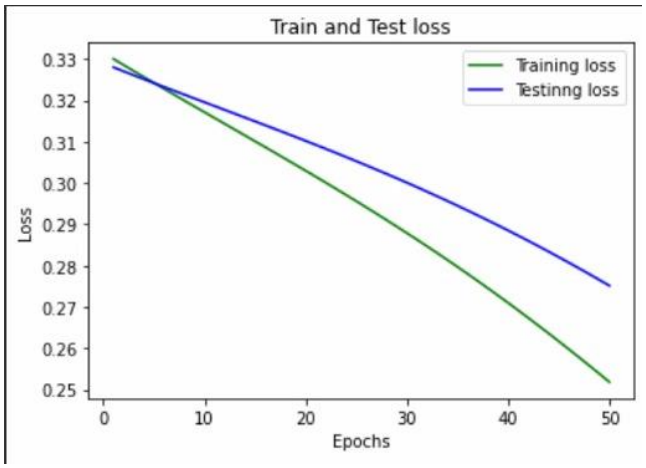


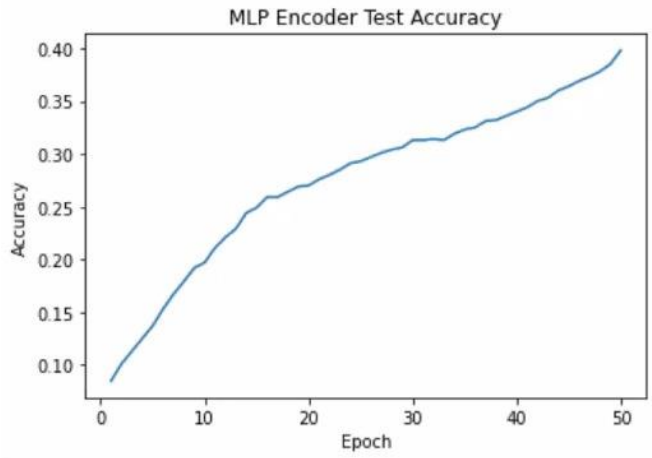
באופן כללי, ישנו טרייד-אוף לגודל ה-latent שנבחר, ככל שיהיה גדול יותר, כך נוכל ככל הנראה לשחזר תמונות בצורה הטובה ביותר כי נוכל לשמור יותר מידע מתמונת המקור. אמנם, כפי שניתן לראות בגרף, הגדלת ה-latent יוצרת קורולציה גבוהה בין חלק מהקואורדינטות. נרצה במודל שלנו שהקורולציה תהיה נמוכה ככל שניתן וכך למקסם את יכולת ותועלת ה-AE, אך עם זאת נרצה לראות תוצאות טובות ושמודל ה-AE יהיה אקפסרסיבי מספיק כדי לבטא את מרחב התמונות. כפי שניתן לראות ולשער, המודל עם latent בגודל 3, בעל הקורולציה הנמוכה ביותר, אך תוצאותיו היו נמוכות (מה שהתבטא ב-loss גבוהה) ולא הצלחנו לשחזר איתו תמונות ברמה גבוהה.

לבסוף, כי לפי גרף זה וה-loss של כל מודל בחרנו ב-latent space בגודל 15.

3) במשימה זו, נדרשנו להשתמש בטכניקת Transfer learning עליה למדנו.
לקחנו את ה-Encoder המאומן, והוספנו מעליו רשת MLP קטנה כדי לפתור את בעיית קלסיפיקציית המספרים מה-Data Set של MNIST.
נראה, כי מאחר וה-Encoder שלנו מאומן ויודע לזהות מאפיינים חשובים במספרים, את משימה זו ביצענו עם מספר דגימות קטן מאוד (כ-100 דגימות אימון ו-80 דגימות test).
מבחינת ארכיטקטורה, הגדרנו את ה-MLP להיות קטנה (לפי ההנחיות), הרשת כוללת שכבה פנימית אחת, עם 128 נוירונים נלמדים סך הכל. בין השכבות הפעלנו ReLU ולבסוף הפעלנו Softmax כדי לסווג את התמונה למספר.
בחרנו להמשיך עם גודל ה-latent space של הסעיף הקודם (15) וכדי לייצג את ה-Labels העברנו את המספרים לייצוג של One-Hot-Encoding (וקטור באורך 10).
לתהליך אימון זה ביצענו 50 epochs.

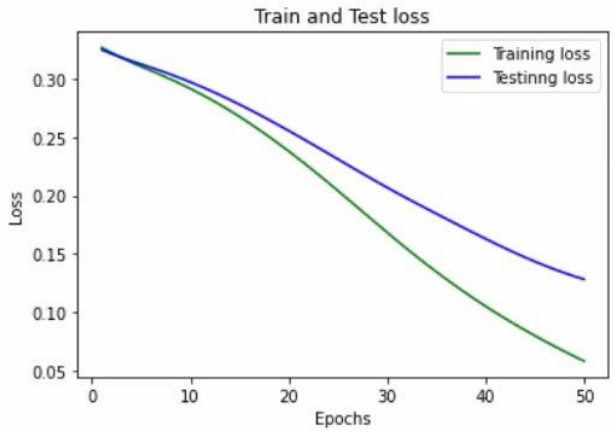
נציג תחילה את תוצאות הרשת מבלי שאפשרנו לה לשנות את משקולות ה-Encoder:

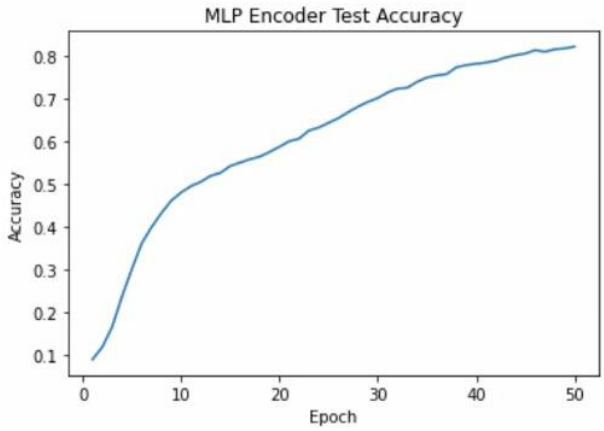




ניתן לראות, כי ה-loss מגיע ל-0.28 וה-Accuracy מגיע ל40%. תוצאות אלה אינן מרשימות במיוחד וניתן לייחס אותן ל-Data הדל עליו אומנה הרשת.

כעת נראה את תוצאות המודל עם Fine-Tuning על משקולות ה-Encoder:





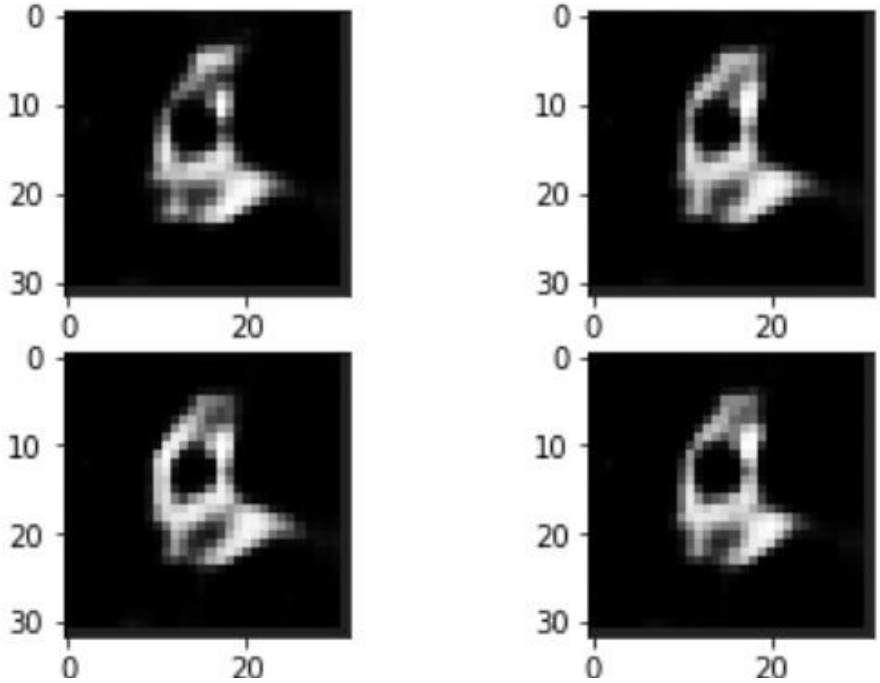
ניתן לראות, כי תוצאות הרשת השתפרו. ה-Loss מגיע ל-0.15 וה-Accuracy ל-80%. שיפור זה מגיע מיכולת הרשת לשפר את משקולות ה-Encoder ולהתאימן למשימת הקלסיפיקציה.

נציין כי לא ביצענו מספר גדול יותר של epochs, כי אחרת היינו מגיעים ל-overfit (במיוחד ברשת השנייה). נמנעו ממצב זה, כי רצינו להשוות את המודלים בצורה נכונה ולא להוסיף משתנים נוספים לבעיה.

4) במשימה זו, נדרשנו לבנות GAN שידע לחקות את מרחב ה- Latent space הנוצר על ידי ה-Encoder המאומן שלנו.
תחילה, כאשר ניגשנו למשימה, השתמשנו בשיטה הקלאסית של בניית ואימון GAN. הגדרנו שתי רשתות המתחרות זו בזו, הראשונה (G) מנסה ליצור דגימות מה-latent space כאשר הזנו לה רעש, והשנייה (D) מנסה להבחין בין דגימות מ-G לבין דגימות אמיתיות מה-Encoder.

נציין כי בארכיטקטורה של הרשתות, דאגנו כי "עוצמתן" תהיה שקולה. אולם, שיטה זו הביאה ל-Mode collapsing כאשר בכל ריצה, המודל G למד להוציא מספר זהה בצורה שונה.

לדוגמה, באחת הריצות, G למדה כי שינויים קטנים במספר 8 מזכירים הרבה מהמספרים והוציאה את הדוגמאות הבאות:



לאחר מכן, בחרנו לשנות גישה והחלטנו לפתור את הבעיה בשיטת ה-WGAN עליה למדנו. לאחר בדיקה מעמיקה בנושא ומעבר על המאמר שהוצג בכיתה, בחרנו תחילה בהיפר-פרמטרים שהוצגו בו:
Batch size = 64, LR = 5e-5, Critic step = 5, Clip value = 0.01.

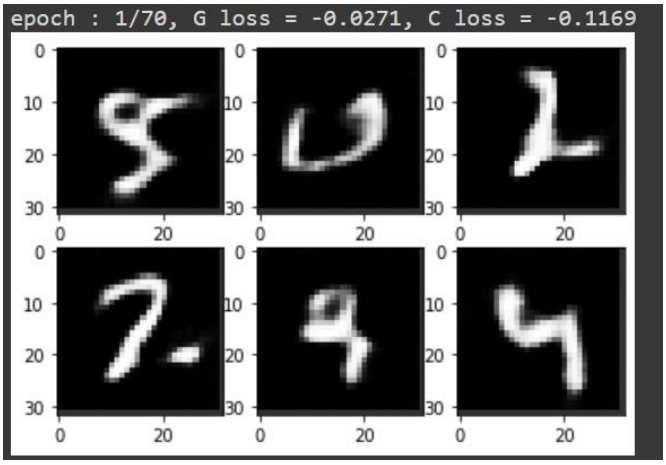
לאחר מספר ניסיונות לא מוצלחים במיוחד, החלטנו לשנות פרמטרים אלו. שמנו לב, כי שינוי היפר-פרמטרים אלו, הביא לתוצאות טובות בהרבה מהניסיונות ההתחלתיים.
לבסוף, ההיפר-פרמטרים שנתנו לנו את התוצאות הטובות ביותר היו:
Batch size = 256, LR = 5e-4, Critic step = 5, Clip value = 0.12.

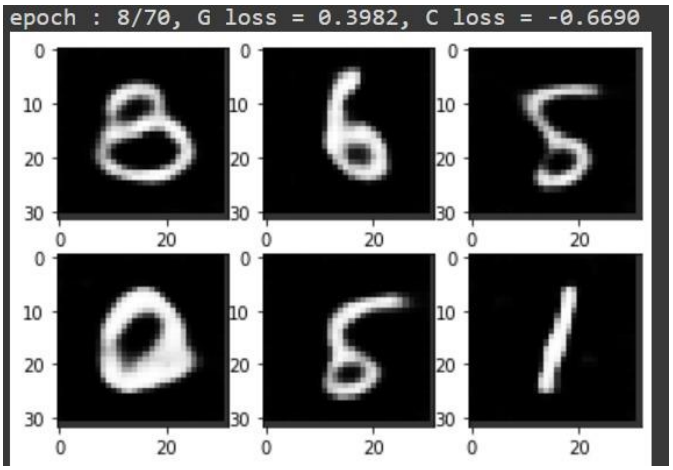
בנוגע למספר ה-epochs אשר ביצענו, ראינו וקראנו כי לוקח זמן רב לאמן את מודל ה-WGAN (ו-GAN בכללי). המודל משתפר במעט בכל epoch לכן בחרנו לבצע מספר גדול יחסית של epochs (70) אך מספר הגיוני מבחינת המשאבים הקיימים לנו.

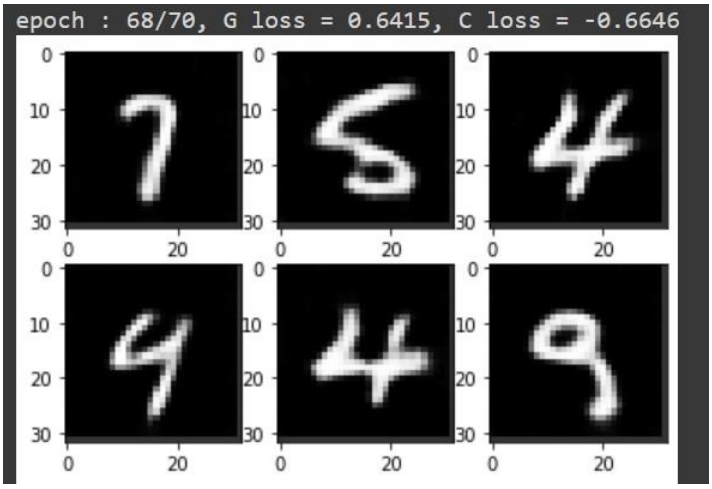
מבחינת הארכיטקטורה של הרשתות, נעזרנו בבחירות הנעשו במאמר, וכללנו במודלים שכבות BatchNorm בנוסף לשכבות הלינאריות. נוסף על כך, בחרנו להשתמש ב-LeakyReLU כפונקציית האקטיבציה שלנו.
G אומנה לקבל וקטור רעש בגודל 10, הנדגם מהתפלגות N(0, 1).

נציג מספר תוצאות שהופקו מהמודל ולאחר מכן נספר תוצאות מאינטרפולציה של וקטורי הקלט.

תוצאות מתהליך האימון:

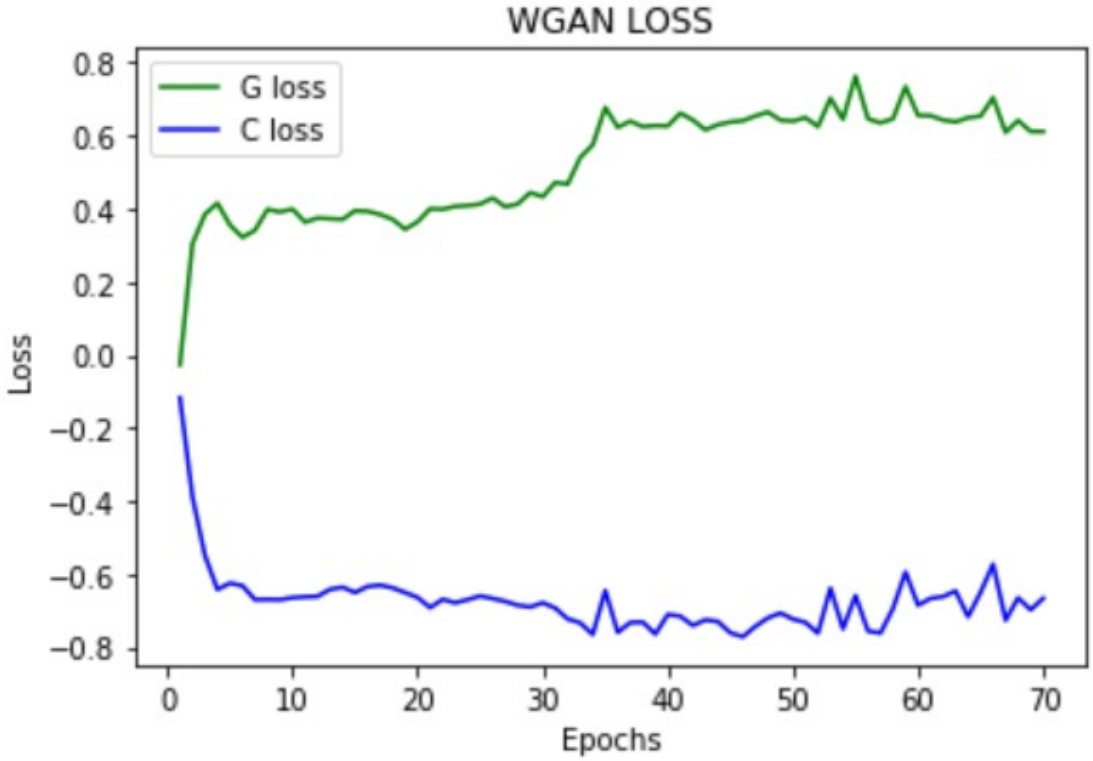






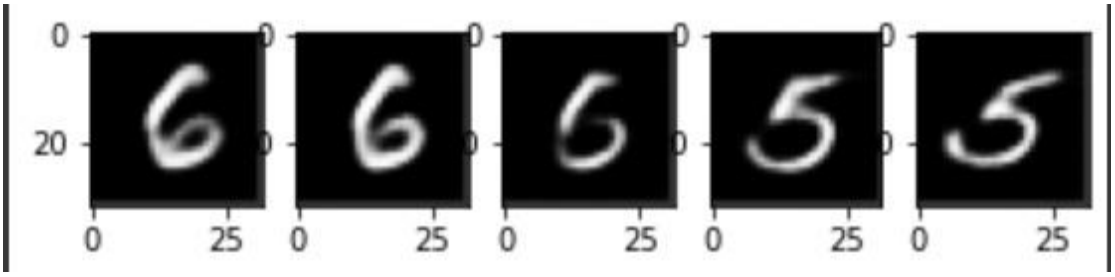
ניתן לראות, כי עם התדמות תהליך האימון, המספרים שה-Decoder פלט מתוצאות G ברורים יותר, ול-Critic קשה יותר להבדיל בינם לבין מספרים אמיתיים מ-MNIST.

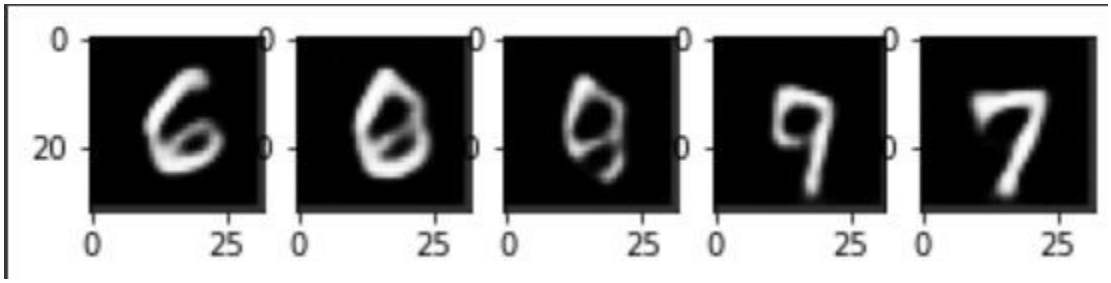
נציג את Loss המודלים:



ניתן לראות מהגרף, כי ה"קרב" בין G ל-C הינו קרב שקול, בו המודלים מאתגרים אחד את השני.

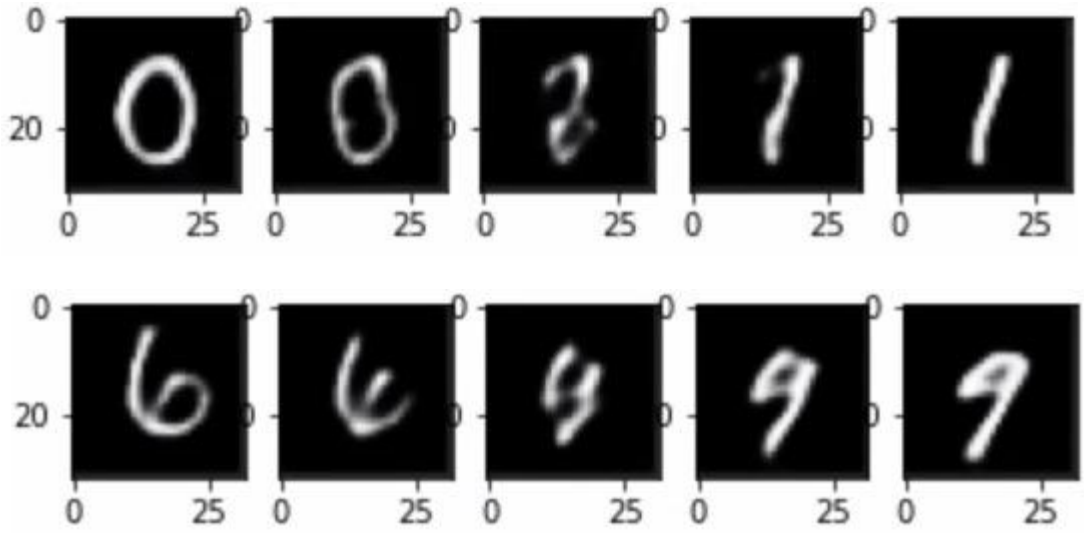
נציג מספר תוצאות אינטרפולציה של G:





ניתן לראות כי תהליך האינטרפולציה ב-G מתרחש כמצופה. מתכונות ה-GAN, עליו להוות מרחב קומפקטי. כלומר לכל z לקיים G(z) = X, עבור X ממרחב המדגם. אכן ניתן לראות, כי לכל z בתהליך האינטרפולציה, פלט ה-latent vector שיצר G, הביא לתמונת מספר מה-Decoder.
באינטרפולציה הראשונה, בין 6, ל5, ניתן לראות כי גם בשלב האמצעי, פלט G הביא לספרה 6 ולא לספרה משולבת (ולא קריאה) בין 5 ל-6.
באינטרפולציה הראשונה, בין 6, ל7, ניתן לראות כי בשלבי האמצע, G יצרה latent vectors שהביאו לספרות 8 ו-9 ושמרה על תכונת הקומפקטיות.

נציג כעת את תוצאות ה-AutoEncoder על אינטרפולציה בין מספרים:

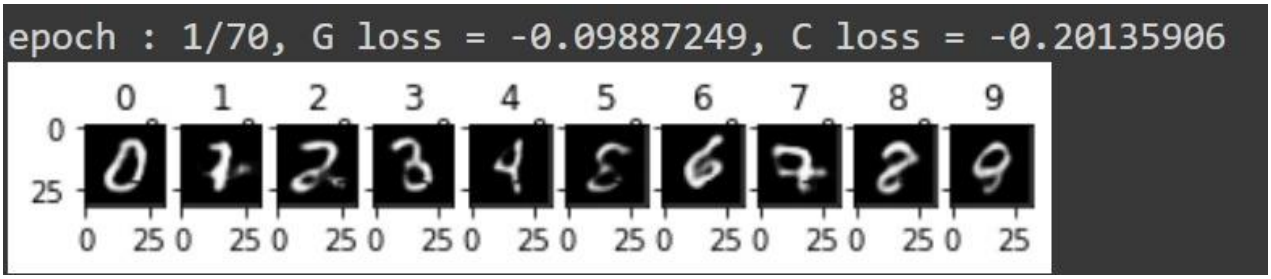


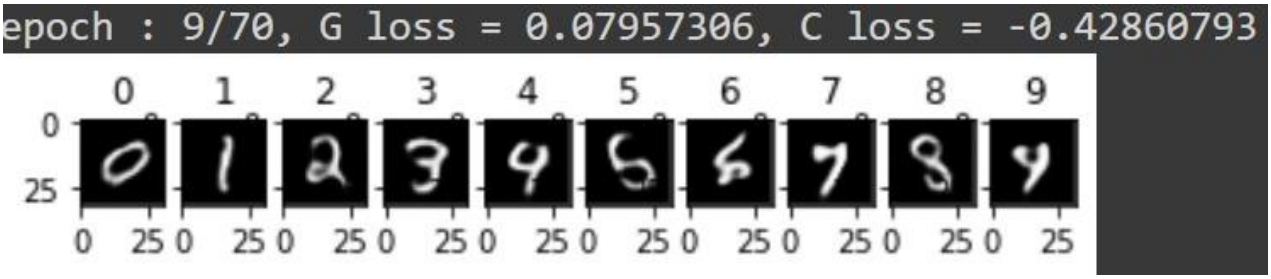
ניתן לראות, כי בניגוד ל-WGAN, תהליך האינטרפולציה ב-AE שונה במעט. נשים לב, כי במהלך התהליך, נקבל תמונות של מספרים לא קיימים (שילוב של המספרים ההתחלתיים). בדוגמה הראשונה, התמונה באמצע, הינה שילוב של 0 ו-1 ואינה מהווה מספר ובצורה זהה בדוגמה השנייה השילובים בין 6 ל-9 יוצרים מספרים לא קריאים.

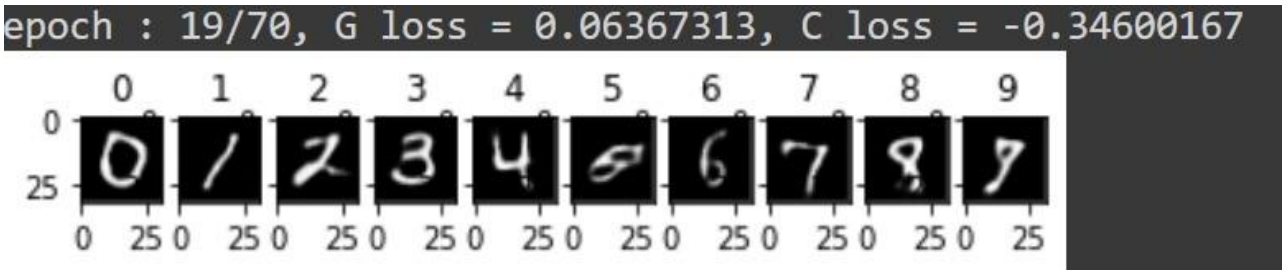
5) במשימה זו, נדרשנו לשפר את מודל ה-WGAN שיצרנו, כך שכעת נוכל לבחור מספר אותו נרצה לייצר והרנדומליות תבוא לידי ביטוי בצורתו השונה.
למשימה זו, בחרנו להפוך את מודל ה-WGAN שלנו ל- Conditional WGAN. שרשרנו לאינפוט של G ול-Critic וקטור באורך 10 המייצג את label הספרה הרצויה (OHE).
כך למעשה, ניצור תלות בין מספר לבין ה-latent vector הרצוי, והרעש שאותו נוסיף ישפיע רק על צורת הספרה.

למעשה, זהו השינוי היחידי אותו ביצענו בסעיף זה מהארכיטקטורה שהוצגה לעיל.

נציג מספר מתוצאות המודל בתהליך האימון:



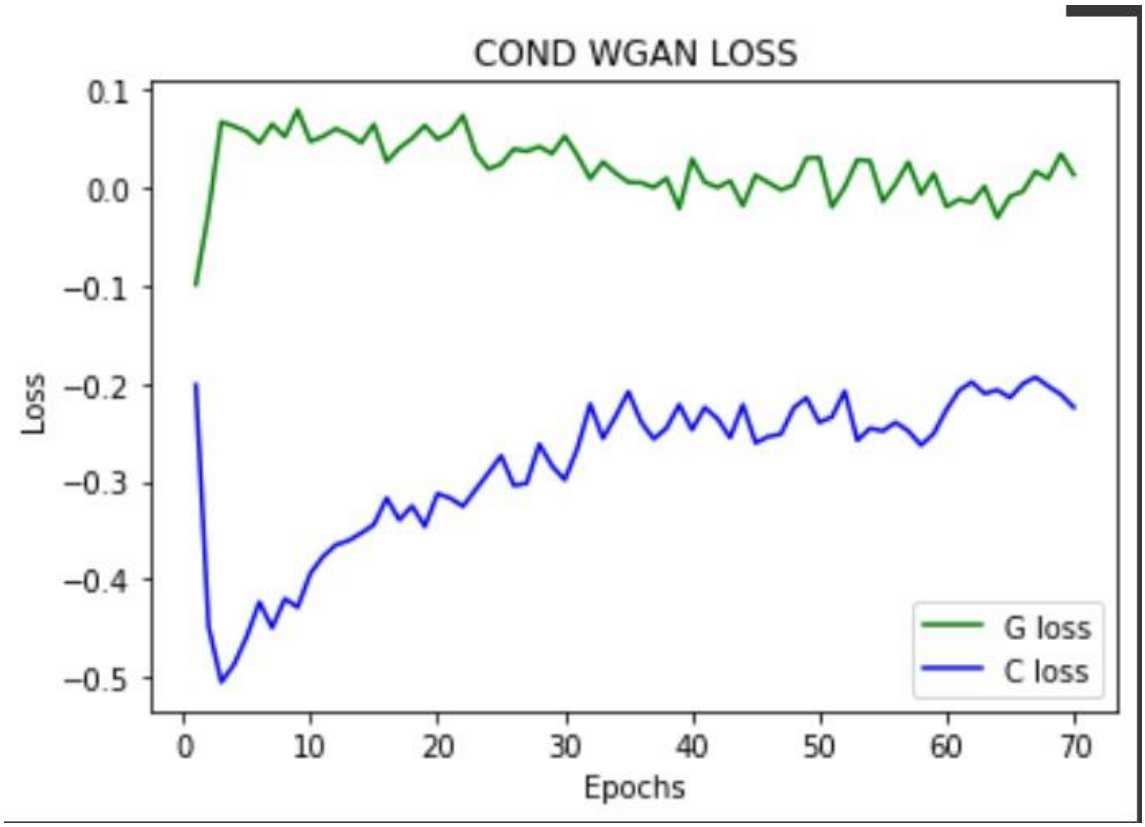




כפי שניתן לראות, כבר לאחר 20 epochs תוצאות המודל היו מרשימות.
הדבר אשר היה חסר למודל בשלב זה, הינו היציבות. נדגים איך מספר ה-epochs הגבוהה תרם ליציבות המודל לקראת סוף האימון שלו.

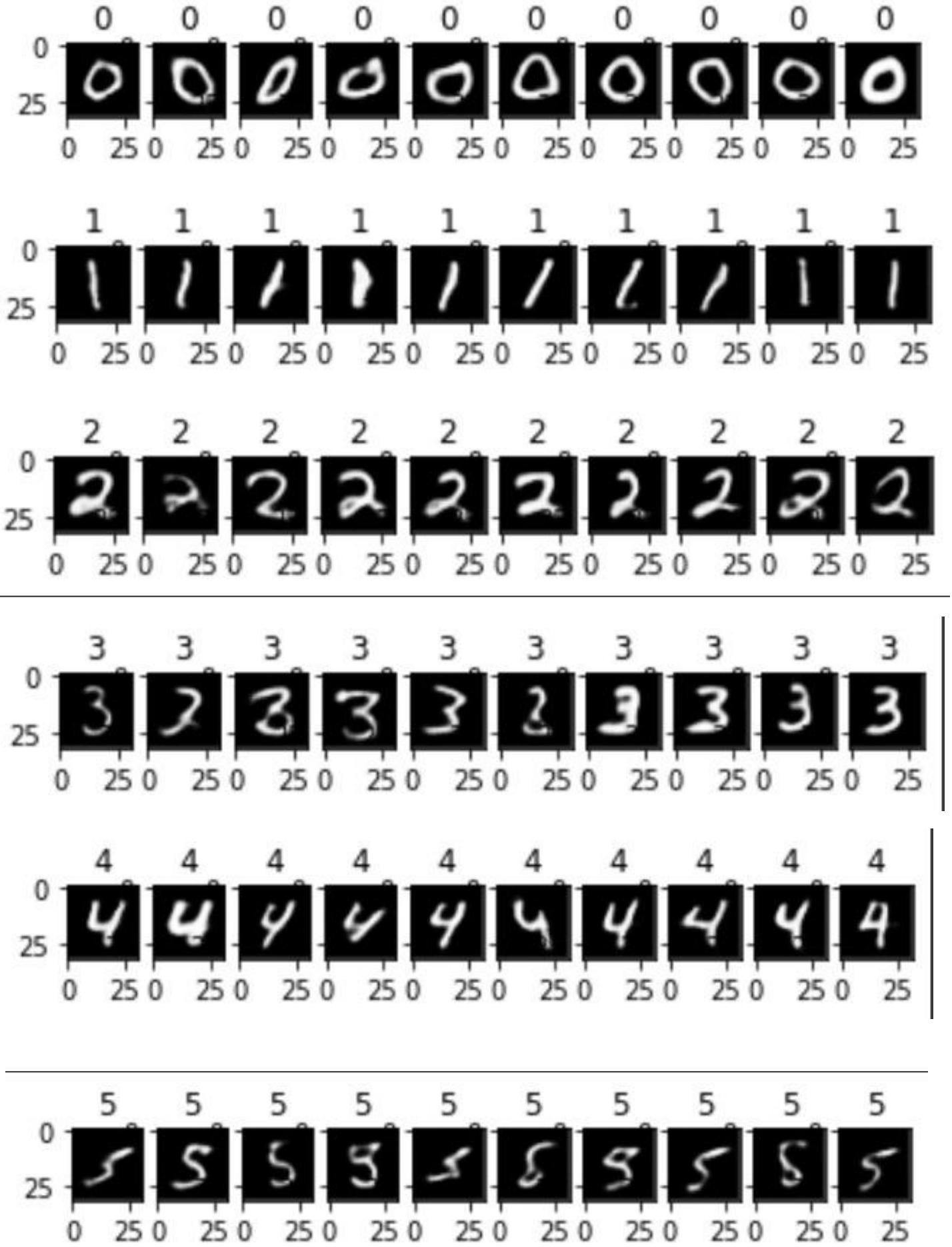
נציג את תוצאות 5 ה-epochs האחרונים:

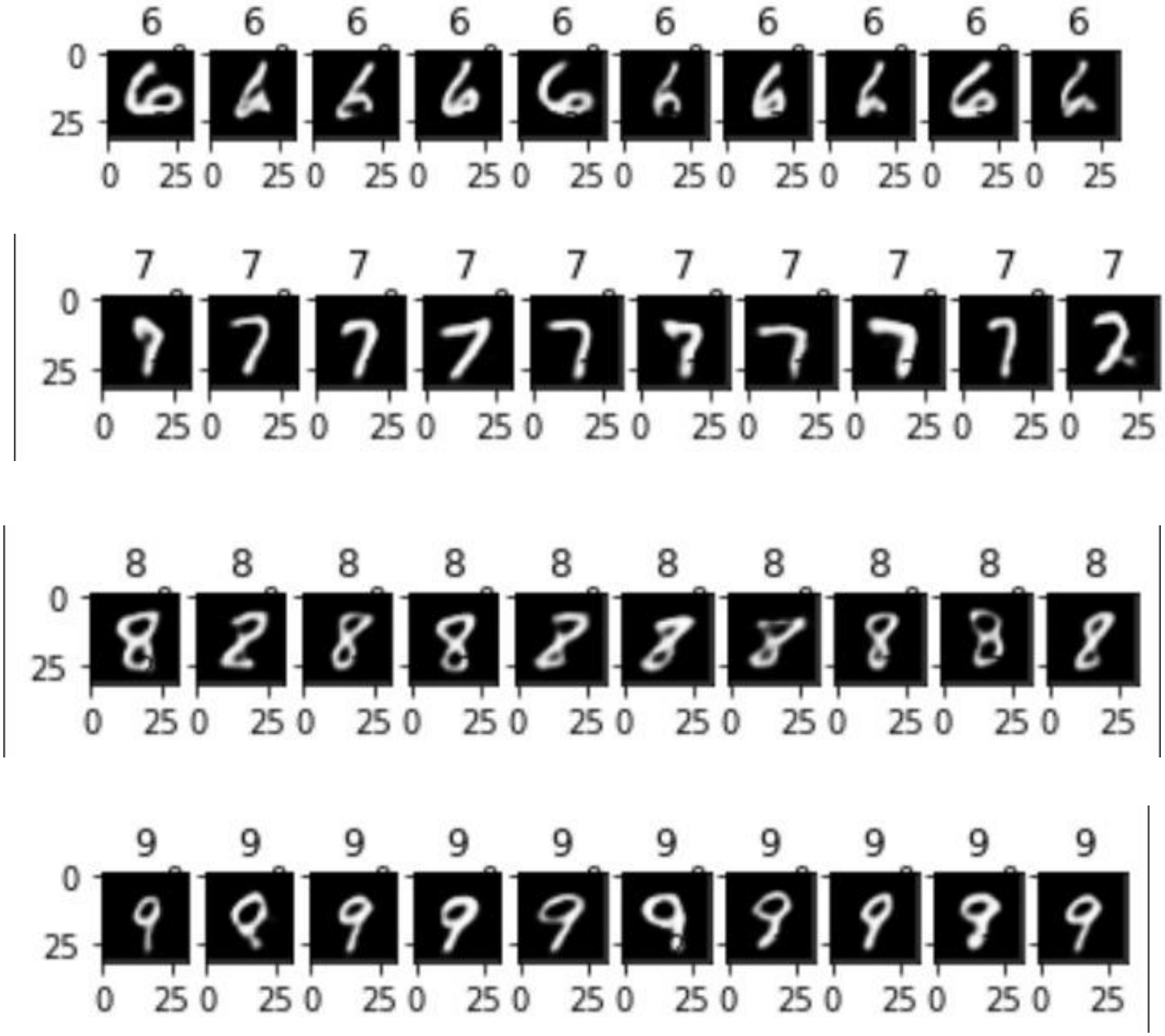
נציג את loss המודל:



מודל זה, בלי קשר למשימתו ליצור תלות בין הספרה הרצויה לפלט, הביא לתוצאות טובות יותר מהמודל הקודם (דבר אשר בא לידי ביטוי בתמונות הפלט שה-Decoder יצר על פלטיו).

נדגים את הצורה בה המודל מתייחס לרעש וכיצד הוא משפיע רק את סגנון המספרים. נציג לכל ספרה, 10 הרצות שונות של המודל עם רעשים שונים –


0 0 0 0 0 0 0 0 0 0
1 1 1 1 1 1 1 1 1 1
2 2 2 2 2 2 2 2 2 2
3 3 3 3 3 3 3 3 3 3
4 4 4 4 4 4 4 4 4 4
5 5 5 5 5 5 5 5 5 5
0 25


6 6 6 6 6 6 6 6 6 6
7 7 7 7 7 7 7 7 7 7
8 8 8 8 8 8 8 8 8 8
9 9 9 9 9 9 9 9 9 9
0
25
0 25

חלק תאורטי:

1) Convolution-

נרצה לחקות את אופן פעולת קונבולוציה, באמצעות ארכיטקטורת BERT. נזכיר כי קונבולוציה, הינה פעולה שאין לה את תכונת הזיכרון שיש ל-BERT והיא LTI.

נבחר להזין את התמונה לרשת BERT באופן הבא, נשטח את התמונה ונוסיף לכל פיקסל את מיקומו המקורי בתמונה (2D encoding). באופן זה, הרשת תתייחס לכל פיקסל כאל מילה. בנוסף, הטמעת המיקום תאפשר לארכיטקטורת BERT לקשר בין פיקסלים קרובים (כמו פעולת קונבולוציה) בצורה דומה לדרך בה היא מקשרת בין מילים קרובות במשפט. שימוש BERT ב-Multi-Headed-Self-Attention דומה לשימוש ב-Multiple kernels של רשת קונבולוציה הפולטת מספר רב של ערוצים. נציין, כי מודל זה יכול לחקות את פעולת הקונבולוציה, אך פוטנציאלית, יכול להשיג הרבה מעבר, שכן בשונה מפעולת קונבולוציה, יכול לקשר בין אזורים רחוקים בתמונה. אם נרצה לכפות על מודל זה, התנהגות דומה יותר לפעולת הקונבולוציה, נוכל להשתמש ב- Restricted attention כדי למנוע מהמודל לקשר בין פיסקלים שלא נמצאים ב-Kernel.

FC-

נזכור כי FC שקולה לפעולת קונבולוציה עם קרנל של 1*1, לכן נוכל להשתמש בפתרון המוצג לעיל על מנת לחקות את פעולת ה-FC. נגדיר את גודל ה-Restricted attention להיות 0 (ללא תלות באחרים) ואת מספר ערוצי הפולט להיות 1. לבסוף נסכום את וקטור הפלט כדי להשיג פעולה דומה ככל שניתן, לFC.

2) בשאלה זו, לכל ארכיטקטורת GAN, נסביר כיצד הייתה עובדת עם כל אחד מה-Datasets הנתונים.

Regular GAN-

- Dataset 1: מבין ה-datasets הנתונים, זהו ה-dataset המתאים ביותר למודל זה. אולם, גם איתו, לא נוכל לצפות לתוצאות מרשימות במיוחד. מדוגמאות רבות שראינו, ניתן לצפות מ-GAN במקרה זה, להגיע ל-Mode collapsing. זאת מאחר ומשימתו היא רק לפלוט תמונות מהמחלקה B ולא ללמוד ולהבין את הקשר בין המחלקות.
- Dataset 2: בדומה ל-dataset 1, בעיה זו אף תהיה קשה יותר ל-GAN הרגיל ונצפה לתוצאות פחות מרשימות.

- Dataset 3: זו אינה משימה שמתאימה ל-GAN רגיל. חוסר ההתאמה נובע מכך שה-Dataset הינו Unpaired, וללא התאמה בין תמונת קלט לפלט לא נוכל לבצע Image translation כנדרש. ניתן לצפות גם במקרה זה ל-Mode collapsing מצד הגנרטור.

<u>-Cycle GAN</u>

- Dataset 1: המודל אמור להצליח לבצע את המשימה עם Dataset זה. ההתאמה בין המחלקות אמורה לעזור למודל להגיע להעתקת זהות בין המחלקות.
- Dataset 2: המודל אמור להצליח לבצע את המשימה עם Dataset זה. נציין כי ככל הנראה, משימה זו תהיה מורכבת יותר עבור המודל מאשר המשימה הקודמת וניתן לצפות לזמן אימון והתכנסות ארוך יותר.
- Dataset 3: זהו ה-Dataset המתאים ביותר למודל. משימה זו, הינה משימה קלאסית למודל זה (כפי שראינו בהרצאה עם הסוסים והזברות לדוגמה). ה-CycleGAN יודע לעבוד עם Unpaired data וליצור בו העתקת זהות, והוא בעל הסיכוי הגבוה ביותר למצוא התאמה בין המחלקות.

<u>:Conditional GAN</u>

- Dataset 1: מודל זה אמור להצליח בצורה טובה על ה-Dataset הנ"ל, ההתאמה בין המחלקות תעזור לנו בתהליך האימון ונדע להדריך את הגנרטור ללמוד ולהבין את הקשר בין המחלקות.
- Dataset 2: גם ב-Dataset זה, המודל אמור להצליח לבצע את המשימה בצורה טובה. למרות השינויים מה-Dataset הראשון, המודל יוכל ללמוד את משפחת התמונות במחלקה B הקשורות לכל תמונה במחלקה A. בשונה מה-CycleGAN, כאן לא נשאף להגיע להעתקת זהות בין המחלקות ולכן הבדלים מינוריים לא ישפיעו רבות על למידת המודל ונוכל לצפות להתכנסותו.
  נציין כי ככל שחוסר ההתאמה עולה, משימה זו הופכת לקשה יותר עבור המודל.
- Dataset 3: מודל זה אינו מתאים למשימה זו. משימה זו הינה Unsupervised ולכן לא נוכל להכוין את המודל ולהתנות תמונה ממחלקה A בתמונה ממחלקה B.