תרגיל בית 3 - למידה עמוקה

**מגישים:**

הילה ניסטל 205766496

בר יפרח 312425036

ירון טל 205592652

**ארכיטקטורה של האוטואנקודר**

יצרנו את המחלקה Dataloader\_recsys  על מנת לנהל את כל הפונקציות על הדאטה הגולמי , ויצירת הדאטה בצורה שנוכל להכניס לרשת וכל הפונקציות התומכות לתהליך הולידציה והלמידה.

1. הגדרת האנקודר והדיקודר

**ה-encoder** מקבל ווקטור בינארי בגודל של כמות הסרטים ( עבור כל משתמש). בווקטור יש 1 בכניסות של סרטים שהמשתמש ראה ואפס אחרת.

השתמשנו בשכבה נסתרת אחת (לאחר ניסוי ותהייה) וגודלה הוא היפרפרמטר שקיבענו לאחר חיפוש ידני.

השתמשנו בפונקציית אקטיבציה סיגמואיד – ניסינו את הפונקציות tanh , relu אך אלו נתנו תוצאות פחות טובות מאשר הסיגמואיד במקרה הזה.

ה-decoder

מחזיר למימד כמות הסרטים ואז ביצענו סיגמואיד מכיוון שאנו מצפים לערכים שהם 0/1 ופונקציית סיגמואיד מחזירה ערכים בטווח הנ"ל

1. פונקציית loss – השתמשנו בMSE

בנוסף , בחנו את הביצועים של BCE אך התוצאות עם הפונקציה הזו היו דומות לשימוש בMSE.

1. רגולריזציה – ברגולריזציה נחלק את ההתייחסות עבור כל סוג של קובץ:

**סרטים אקראיים (Randomtest)**

הוספנו דרופאוט לפני הכניסה לרשת על מנת להרעיש את הווקטור בכניסה ( מימוש הרעיון של (**Denoising Autoencoders** ולאחר מכן מנסים לחזות את הווקטור המקורי. (בדומה למה שצוין במאמר (Autorec

**סרטים פופולריים ( (Popularitytest**

על מנת לייצר למידה בדגש על הפופלריים - יצרנו ווקטור מסיכה. בכל פעם שאנחנו נכנסים ללולאת האימון , אנו מגרילים ווקטור (**ווקטור ההתפלגות** נקרא לו בהמשך) על בסיס ההתפלגות של הסרטים(לפי רמת הפופולריות) ומכפילים אותו בפקטור שהוא היפרפרמטר של המערכת (על מנת לייצר ערכים הגיוניים בהשוואה למול ההתפלגות האחידה. כלומר, מי שפופולארי הוא בסיכוי גבוה יותר להיבחר). ההיפר פרמטר שבחרנו גרם לכך בערך שבין 5% ל 8% מהסרטים שלא נצפו יוגרלו בכל איטרציה. לאחר מכן , אנו מגרילים ווקטור מהתפלגות אחידה בין 0 ל- 1 (נקרא לו **ווקטור התפלגות אחידה**) בגודל של מספר הסרטים שבעזרתו נבצע סימולציה.

עבור כל הסרטים שהמשתמש ראה ( שמופיעים כ-1 בווקטור של המשתמש) אנחנו מכניסים 1 בכניסה המתאימה בווקטור המסכה. עבור כל שאר הסרטים אנחנו משווים בין הערך הקיים עבורם בווקטור ההתפלגות לבין הערך הקיים עבורם בווקטור ההתפלגות האחידה – במידה והערך בווקטור ההתפלגות גדול יותר אנחנו מכניסים 1 בווקטור המסכה, כל השאר יהיו אפסים. כל סרט שלא נצפה יוגרל לקבל ערך 1 בוקטור המסיכה בסיכוי שפרופורציונלי לרמת הפופולריות שלו(כפול ההיפרפרמטר שבחרנו). בכל וקטור כזה בערך בין 5 ל 8 אחוז מהסרטים שלא נצפו היו "דלוקים". לסרטים פופולריים שלא נצפו היה סיכוי גדול בהרבה מסרטים לא פופולריים שלא נצפו להיות דלוקים.

לאחר ביצוע ה- forward אנחנו מכפילים את ווקטור התוצאה של ה- forward בווקטור המסכה שלנו על מנת שה LOSS יחושב רק על חלק מהאיברים בוקטור האאוטפוט בדגש על הסרטים שנצפו על ידי היוזר ועל הסרטים הפופולריים(שסביר שניתקל בהם בקובץ הטסט הפופולרי). הדבר הזה גורם לכך שיש נטייה חזקה יותר ללמידה עבור פופולריים מכיוון שהוא מאפס כניסות של סרטים לא פופולריים בהסתברות גבוהה יותר וכך מאפשר לרשת לתקן באופן מובהק יותר לכיוון הפופולריים.

בנוסף , עבור 2 הסטים השתמשנו בweight decay

על מנת להחליש את מקדם הלמידה בחלוף האפוקים.

1. **הצדקת הארכיטקטורה**

בהתחלה ניסינו כמה שכבות נסתרות וראינו שאין שיפור משמעותי , לעומת האטה משמעותית בזמן האימון וכמות פרמטרים גדולה יותר אז בחרנו בשכבה נסתרת אחת.

הוספנו dropout לאחר שראינו שמביא לביצועים יותר טובים.

כמו כן , גם גודל השכבה נקבע לאחר חיפוש ידני.

**תיאור תהליך הלמידה**

1. **חלוקה לאימון\ולידציה**

יצרנו סט וולידציה בכך שהסרנו סרט אחד שהמשתמש ראה מהסט של האימון ואז בכל פעם שעשינו inference  הגרלנו למולו סרט שהמשתמש לא ראה וכך חישבנו את מדד

ה- Accuracy.

בוולידציה , במקרה הרנדומי , הגרלנו סרט שהמשתמש לא ראה באופן אקראי.

במקרה הפופולרי , הגרלנו סרט שהמשתמש לא ראה לפי התפלגות הפופולריים – כלומר סרט שהוא יותר פופולרי יוגרל בסיכוי גבוה יותר ביחס לאחרים(כמות הצפיות בו חלקי סה"כ כמות הצפיות של כל היוזרים בכל הסרטים).

1. **תהליך העדכון (update scheme)**

גודל הבאטצ' שלנו היה 1. בכל איטרציה אנחנו מבצעים את ה- forward ומתקנים לפי הלוס MSE באמצעות אלגוריתם ה backpropogation . בחלק הרנדומי התיקון היה אחיד ופשוט בעוד בחלק הפופולרי התיקון היה בדגש על הסרטים שנצפו ועל הסרטים הפופולריים(ע"י שינוי הוקטור כפי שצוין לעיל).

כל מעבר על הווקטורים של כלל המשתמשים נקרא אפוק.

**איך מקבלים החלטות במודל?**

בסוף תהליך האימון (ב-2 השיטות) יש לנו ווקטור פלט(של האוטואנקודר) עבור כל משתמש המכיל ערכים בגודל מספר הסרטים. בהשוואה של 2 סרטים אנחנו נעדיף את זה עם הערך הגדול יותר בווקטור הפלט וכך נתעדף סרט על פני משנהו.

**הבדלים בין הרנדומי לפופלרי**

עבור הסט הרנדומי הגענו לדיוק של 0.932

עבור הסט הפופולרי הגענו לדיוק של 0.87

התוצאה היא הגיונית – ציפינו לקבל תוצאה נמוכה יותר בפופלרי מכיוון שה"תחרות" בסט הפופלרי קשה יותר מהרנדומי. ברור שיהיה קשה יותר להפריד בין סרט שיוזר מסוים צפה בו לסרט כמו הארי פוטר לעומת אותו הסרט מול סרט גרמני משנות ה60 .

**תובנות ומסקנות**

שימוש בAutoencoder כמערכת המלצה הוא דבר חדש שבהתחלה היה לנו קשה להתחבר לרעיון. אך ככל שהעמקנו בתרגיל ראינו שזה למעשה דבר הגיוני מאוד , והיכולת של הAutoencoder לחלץ את הסאבטקסט היא לא רעה בכלל ואפשר ליישם את הטכניקה הזו בהרבה תחומים כמו טקסט, וידאו וכו'. היכולת העיקרית החשובה היא היכולת ללמוד רפרזנטציות של דאטא ממימד גבוה באופן לא לינארי(דבר שטרם נתקלנו בו עדיין בקורסים קודמים) ובשיטות המלצה אחרות שפועלות באופן לינארי כמו Matrix factorization .