**רטוב 2 – NLP**

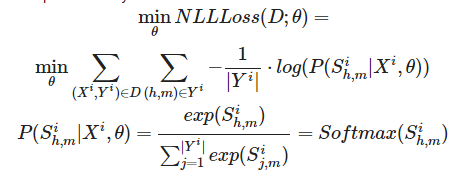
מגישים:

אמיר בלדר – 204179659

רועי (Roy) גנץ – 204506349

**אימון:**

תחילה נתאר מספר גורמים שהיו משותפים עבור שני המודלים שמימשנו.  
פונקציית ה-LOSS:  
את פונקציית ה-LOSS מימשנו בשונה מהמאמר שהיה נתון, ובהתאם להוראות שקיבלנו בתרגיל עצמו ובסדנת ההכנה לתרגיל.   
בשביל המימוש נעזרנו בטבלת "אנרגיה":



אם נסתכל על השורה האחרונה בהגדרת הפו', את המונה עלינו לחשב בעזרת הזוגות של head, modifier האמיתיים שקיבלנו כקלט עבור המשפט, ואת המכנה עלינו לחשב עבור כל זוג חוקי של head, modifier שיכול להיות בתוך המשפט.  
אם כן, שמרנו טבלת "אנרגיה" שהכילה את עבור כל אופציה אפשרית במשפט, כאשר השורה הראשונה בטבלה היוותה את הצמדים האמיתיים של head, modifier שניתנו לנו כקלט.  
השימוש בטבלה עזר לנו לבצע את החישוב של פו' ה-LOSS בצורה יעילה יחסית, שכן יכולנו לבצע פעולות וקטוריות על עמודות ושורות הטבלה.

בנוסף, התחשבנו בכך שיהיו מילים שלא יופיעו בקורפוס האימון אך כן יופיעו בבדיקה. כאשר אנו מעבדים את הקלט עבור האלגוריתם כולו, אנו מבצעים ספירה של מספר הפעמים שכל מילה הופיעה, במעין מילון. כאשר מילה ספציפית לא הופיעה אנו נותנים לה בעצם ייצוג מינימלי באותו המילון המדובר, על מנת שההסתברות לקבלת מילה זו לא תהיה אפס.  
  
המודל הבסיסי –   
מימוש המודל הבסיסי שלנו נעשה בהתאם למאמר:

Simple and Accurate Dependency Parsing Using Bidirectional LSTM Feature Representations

נציין כי מימושו של מודל זה נמצא בקובץ basic\_model.py,  
ונתאר בקווים כלליים את המימוש שלנו:

במודל הבסיסי נאסר עלינו להשתמש ב-embedding מאומן מראש, ולכן היה עלינו להחליט בעצמנו כיצד לבצע את ה- embedding.   
אנו חשבנו ששווה יהיה להפריד בין ה- embeddingשל המילים עצמן (כלומר ה-TOKEN) לבין ה- embeddingשל ה-POS (TOKEN POS). זאת משום שמדובר בשני סוגי מידעים שונים, ועל כן חשבנו שכדאי שהייצוג שלהם ושהלמידה על הייצוג המיטבי שלהם תתרחש בנפרד.  
על כן, לכל אחד מהם הייתה שכבת embedding משלו, כך שהייצוגים (המשקלים) הנכונים לכל אחד מבין שני סוגי המידעים האלה יוכל להלמד בנפרד, ולאחר מכן ביצענו חיבור (concatenate) של הייצוגים שקיבלנו עבור השלב הבא במודל שלנו.

לאחר מכן, בהתאם למאמר, הייצוג המאוחד נכנס כקלט ל-BiLSTM, כאשר הפלט של ה-LSTM היווה את הקלט לשכבת MLP.  
  
לא עשינו כוונון היפר-פרמטרים עבור מודל זה, אלא לקחנו את אלה שהשתמשו בהם במאמר:

MLP\_HIDDEN\_DIM = 50

WORD\_EMBEDDING\_DIM = 100

POS\_EMBEDDING\_DIM = 25

HIDDEN\_DIM = 125

LEARNING\_RATE = 0.01

המודל המשודרג –

היה עלינו לשפר את המודל הבסיסי, והיו לנו מספר רעיונות לשם הביצוע, נסביר את הרעיונות ואת המוטיבציה לביצוע של כל אחד ואחד מהם. היו שתי אבני בוחן כלליות, הדיוק וכמה זמן יקח למודל להתאמן, מתוכן המבחן שעניין אותנו הכי הרבה היה כמובן מבחן הדיוק על ה-trainset.

1. כוונון היפר-פרמטרים – רצינו לעשות כיוונון היפר-פרמטרים בעצמנו. היפר-פרמטרים הינם משהו שמתאים לא רק למודל ספציפי, אלא גם לבעיה ספציפית. כלומר שבמקרה שלנו, עם מאגר האימון והבדיקה שלנו, יתכן מאוד שנקבל היפר-פרמטרים אופטימליים שונים מאשר אלה שהשתמשו בהם במאמר המקורי. על כן, תחילה עשינו כיוונון היפר-פרמטרים עבור הבעיה שלנו, ולאחר מכן ביצענו את כל השינויים האחרים.

בעצם תוצאות הדיוק שלנו לאחר כוונון זה היוו את ה-ground truth עבורנו, וכל שינוי שבא לאחר מכן נבדק אל מול תוצאות דיוק אלה.

1. העמקת הרשת – כפי שלמדנו בהרצאות בקורס, במודלים רבים ב-NLP גילו שהעמקת המודל, מבחינת מספר השכבות שלו, וכן הגדלת נפח המידע שעימו המודל מאומן, משפרים מאוד את התוצאות של המודלים השונים. לדוגמא, BERT מהווה בעצם שרשור של מספר transformer encoders שהיו במודל של Transformer, והגיע לתוצאות יפות. לאחר מכן יצא BERT מורחב, שהינו אותו BERT רק בעל שכבות רבות יותר שמאומן על קורפוס גדול יותר, ואכן הגיע לתוצאות טובות יותר מקודמו. במקרה שלנו, מאגר המידע הינו קבוע, ולא בר שינוי. על כן, חשבנו לבצע את ההעמקה של המודל על ידי שינוי מספר שכבות ה-BiLSTM. כמובן ששינוי מעין זה ישנה את זמן האימון של המודל.
2. שימוש ב-embedding מוכן מראש – שימוש ב-embedding מוכן מראש בעצם יהפוך את החלק הראשון של הרשת שתיארנו בסעיף הקודם למיותרת, אך קראנו ביקורות שונות, וגילינו ששימוש בו יכול להיטיב מאוד עם המודל שלנו. צוואר בקבוק אחד שתיארנו בחלק של העמקת הרשת היה שאיננו יכולים לאמן את המודל שלנו על מאגר מידע עשיר יותר ממה שקיבלנו בתרגיל, והשימוש ב-embedding מוכן בעצם פתר לנו חלק מבעיה זו. מדובר ב-embedding שאומן על מאגרי מידע גדולים מאוד, ולכן הוא מהווה עבורנו בעצם סוג של transfer learning ממודלים אחרים ועשירים יותר משלנו. השתמשנו בסופו של יום ב- Vocab, שהינו embedding מפורסם שגם אושר להשתמש בו בתרגיל זה.
3. שימוש ב-dropout - כפי שתיארנו בשיפור הראשון, התאמנו את היפר-הפרמטרים שלנו ישירות לבעיה הנוכחית שעליה אנו עובדים. גם קיבלנו תוצאות מאוד יפות, ואפילו היו batch-ים שקיבלנו עליהם דיוק של 100%. זה גרם לנו לחשוש מעט מ-overfitting. נציין כי לא ראינו בגרפים שלנו מצב שבו השגיאה על סט האימון ממשיכה לרדת בעוד שהשגיאה את סט הבדיקה עולה, מה שהמווה את הסימון הקלאסי ל-overfitting. אולם עדיין חששנו, שכן אם סט האימון וסט הבדיקה נלקחו ממאגר מידע דומה, יתכן שגם לא נראה את הסימן הקלאסי מתממש. על כן, חשבנו להוסיף dropout, שהינו כלי מוכר להתמודד עם בעיה זו.

1. פונקציית אקטיבציה – מעבר לפונקציית Relu – רצינו לנסות פו' אקטיבציה שונה מאשר המאמר, בשל העובדה של-relu ישנן תוצאות טובות במאמרים שונים.

בסופו של יום, כל השינויים נבחנו לפי מבחן הדיוק, ועל כן נבחרו השינויים: **XXX**

תוצאות שני המודלים:

עתה נתאר את זמני הריצה ואת הדיוק עבור שני המודלים:  
הבסיסי:

המשודרג:

**חסר בחלק זה – זמני הריצה + הגרפים, ולכמה אחוזי דיוק הגענו בשני המודלים בסוף, אבל זה יגזר מהגרפים עצמם.**

**הסקה:**

בחלק זה השתמשנו בקובץ שניתן לנו בתרגיל של chu\_liu\_edmonds ללא שינוי. השתמשנו בתוצאותיו על מנת לבדוק את הדיוק של המודלים שלנו.

מבחן:

להלן התוצאות על סט המבחן כפי שהתקבלנו על ידי שני המודלים שלנו:  
המודל הבסיסי:

המודל המשודרג:

**חסר בחלק זה – זמני הריצה + הגרפים, ולכמה אחוזי דיוק הגענו בשני המודלים בסוף, אבל זה יגזר מהגרפים עצמם, וצריך לכתוב משהו על ממה ההבדלים נובעים וכו'... זמן הריצה יכול לנבוע מההבדלים במבני המודלים, אחוזי הדיוק גם...**

**תחרות:**

כאן נצטרך לכתוב במה בחרנו, אז עוד חזון למועד...