NLP - תרגיל בית 2 - רטוב

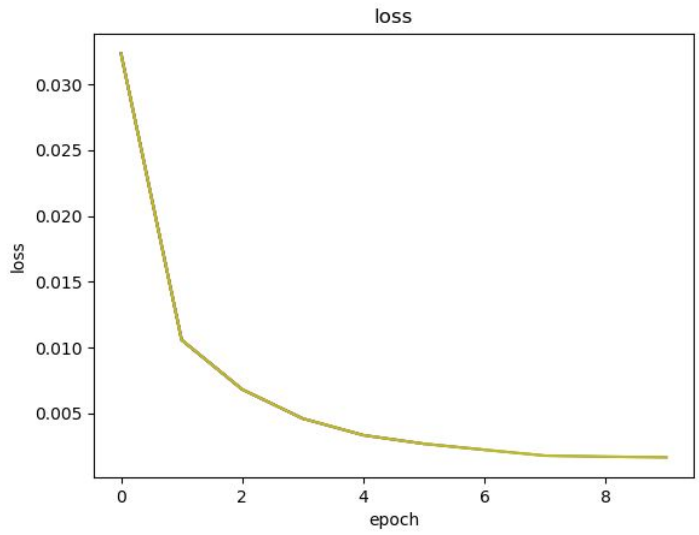
דו"ח מסכם

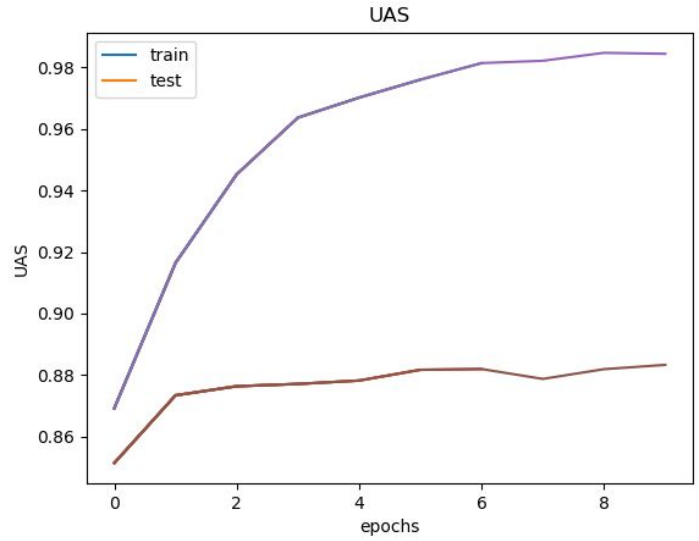
רוברט ספקטור - 305320400 , עופרי אולשביצקי - 311287791

אימון

מודל בסיסי

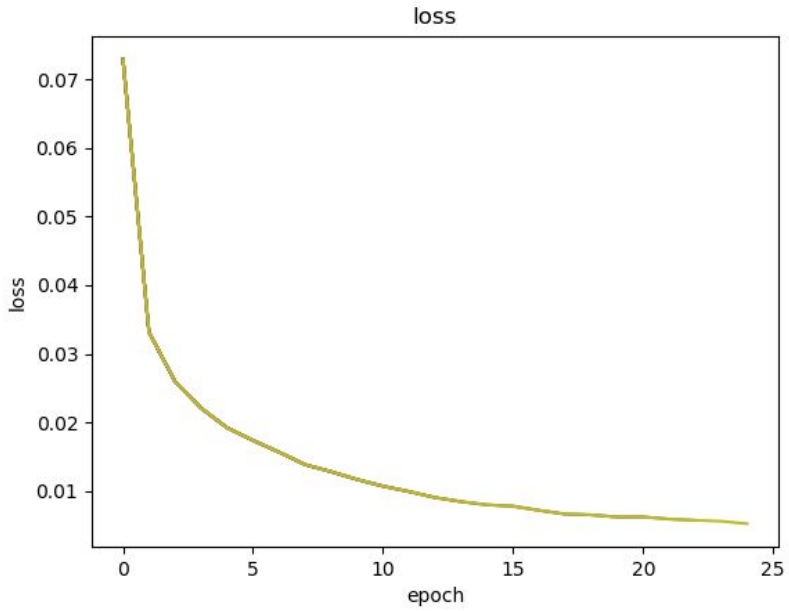
בהסקה על סט האימון עם המודל הבסיסי קיבלנו אחוז דיוק UAS של 98.1%.
את המודל הגרפי הבסיסי מימשנו כפי שהוא הוצג במאמר של Kiperwasser and Goldberg,
זה כולל את כל שכבות הרשת, הערכים של ההיפר-פרמטרים, שימוש ב word dropout ושימוש ב loss augmented inference. השתמשנו בפונקציית לוס CrossEntropyLoss מספריית torch.
מימוש הרשת נמצא בקובץ basic_model.py. האימון בוצע לאורך 10 אפוקים עם באצ'ים של 50 משפטים. זמן האימון לקח 13 דקות (עם חישוב UAS) על המכונה שהוקצתה לנו. להלן הגרפים הרלוונטיים:

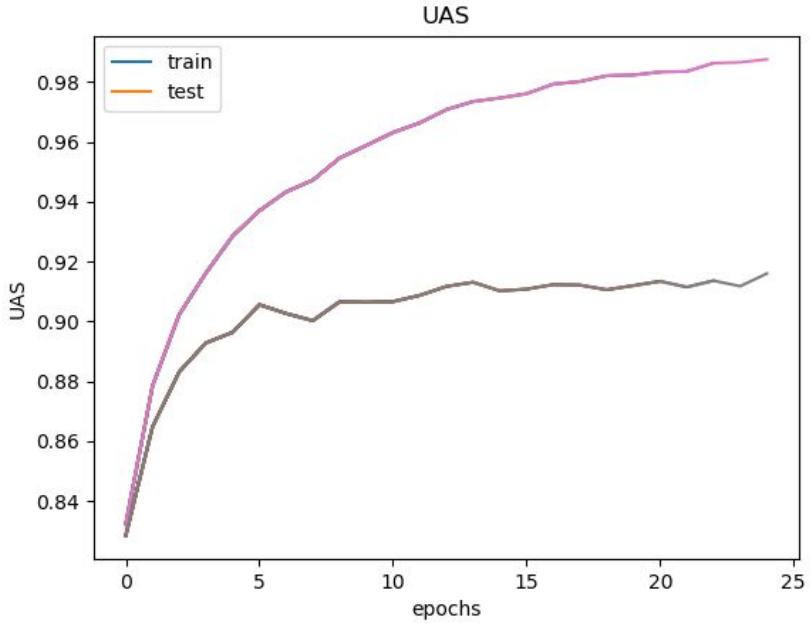




מודל מתקדם:

בהסקה על סט האימון עם המודל המתקדם קיבלנו אחוז דיוק UAS של 98.8%.
המודל מומש עם ארכיטקטורת רשת שונה מהמודל הבסיסי ועל כך נפרט בהמשך. האימון בוצע לאורך 25 אפוקים עם באצ'ים של 25 משפטים. זמן האימון לקח 48 דקות (כולל חישובי UAS) על המכונה שהוקצתה לנו. להלן הגרפים הרלוונטיים:





שיפורים שהכנסנו למודל המתקדם:

- החלפת שכבת הMLP בשכבה אפינית דו-צדדית (Biaffine). יתרון אחד הוא שמדובר במודל פשוט יותר - אין צורך לעבור על כל זוג מילים לאחר הפעלת LSTM. במקום זאת מדובר במכפלה והוספה של מטריצות ווקטור. יתרון נוסף והמשמעותי ביותר הוא שכעת המודל מבחין בין head ל-modifier. נשים לב של-head יש ייצוג יחודי (הגורם שמוכפל משמאל) ול- modifier ייצוג יחודי (הגורם שמוכפל מימן).
נשים לב שהייצוג הזה לא היה קיים במודל הבסיסי כי בשכבות הMLP הייצוג של head ו modifier היה זהה לחלוטין. ייצוג זה הינו ייצוג מדויק יותר של הבעיה. ולכן נצפה שנקבל תוצאות מדויקות יותר.

$$\mathbf{h}_i^{(arc\text{-}dep)} = \mathrm{MLP}^{(arc\text{-}dep)}(\mathbf{r}_i) \tag{4}$$

$$\mathbf{h}_j^{(arc\text{-}head)} = \mathrm{MLP}^{(arc\text{-}head)}(\mathbf{r}_j) \tag{5}$$

$$\mathbf{s}_i^{(arc)} = H^{(arc\text{-}head)} U^{(1)} \mathbf{h}_i^{(arc\text{-}dep)} + H^{(arc\text{-}head)} \mathbf{u}^{(2)} \tag{6}$$

נלקח מהמאמר [deep biaffine attention for neural dependency parsing]

- העמקת רשת הLSTM - העמקנו את הרשת לשלוש שכבות במקום שתיים. ככל שהרשת עמוקה יותר כך המודל לומד להכליל טוב יותר אבל הסיכון ל overfitting עולה, הדרך הטובה ביותר לבדוק זאת היא בעזרת ניסוי וטעייה. מצאנו שאכן עליה לשלוש שכבות שיפרה את ביצועי המודל אבל עליה בשכבה נוספת גרמה ל overfitting גבוה וביצועים טובים פחות.

- הפעלת MLP עם אקטיבציית ReLu על הפלט מהLSTM . הרעיון כאן הוא שה MLP משמש כרגולריזטור ש"קוטע" מידע מיותר שיצא מה-LSTM. אקטיבציית הReLU משמשת בדיוק לשם מטרה זו של קיטוע. הMLP במודל הזה הוא משמעותי מהסיבה שאנו משתמשים ב embeddings במימדים גבוהים יותר ולכן נרצה שכבה ממתנת בין הLSTM ל-Biaffine - על כך נפרט בהמשך. בתמונה למעלה ניתן לראות את את המעבר מLSTM ל score של קשת שהשתמשנו במודל.

- שימוש בDropping. במהלך האימון אנו מבצעים word dropping בייסיאני אגרסיבי שמחליף מילים ב unknown בהסתברות של ⅓ ,כמו כן אנו זורקים שכבות ברשת הLSTM בהסתברות ⅓.
המטרה כאן היא רגולריזציה. אנו מתאמנים במיוחד לתרחישי אי ודאות ולכן זריקה של מילים ושכבות הרשת מדמה למודל תרחישי אי ודאות ומאמנת את המודל גם על וקטור הunknown.
ואכן במבחן התוצאה הdropping שיפר לנו את ביצועי המודל .

- שימוש ב word embeddings מאומנים מראש - glove.840B.300d . במודל אנו משתמשים בשני סוגי אמבדינגס האחד - בדומה למודל הבסיסי - אמבדינג שמאותחל רנדומלית ונלמד ביחד עם שאר משקולות הרשת. השני - אמבדינג מאומן מראש שמוזן חיצונית למודל והמודל <u>לא</u> נלמד עליו (כלומר לא מחשב עבורו גרדיאנט). בכל כניסה לforward המודל מקבל את שני האמבדינגס , מחבר אותם ולאחר מכן מבצע קונקטינציה עם האמבדינג של ה POS . מכיוון שהמודל לא נלמד על האמבדינגס המאומנים מראש - הייצוג שלהם נשמר. זה מאפשר לנו בעת תהליך ההסקה להזין(חיצונית) את המודל גם באמבדינגס מאומנים מראש של קובץ ההסקה. לכן מילים נדירות שהמודל אף פעם לא ראה עלולות להופיע באמבדינג המאומן מראש ולתת למודל קרבה למילים שהוא ראה בעבר. מכיוון שאנו מחברים שני אמבדינגס, נרצה ששניהם יהיה באותו גודל. האמבדינג המאומן מראש הוא כברירת מחדל בגודל 300 ולכן קבענו גם את האמבדינג הרנדומלי לגודל 300. נציין שניסינו מספר אמבדינגס מאומנים מראש ובחרנו את האחד שנתן את התוצאות הטובות ביותר.

- כיוונון היפר-פרמטרים - איננו יכולים להיות בטוחים שהדאטא עליו Kiperwasser ו Goldberg ביצעו אופטימיזציה דומה לדאטא שבידינו. לכן סביר להניח שהיפר-פרמטרים אחרים יכולים לשפר את הביצועים של המודל שלנו על הדאטא הנוכחי. הכוונון של ההיפר-פרמטרים להלן נעשה באמצעות ניסוי וטעייה.

```
# Hyperparameters
WORD_EMBEDDING_DIM = 300  # must be 300 for external embeddings
POS_EMBEDDING_DIM = 30
BILSTM_DIM = 256
MLP_HIDDEN_DIM = 256
BILSTM_DROPOUT = 0.333
LEARNING_RATE = 0.0025
WEIGHT_DECAY = 1e-5
BILSTM_LAYERS = 3
EPOCHS = 25
accumulate_grad_steps = 25
```

השראה לשינוי ארכיטקטורת הרשת נלקחה מהמאמר של Timothy Dozat ו Christopher D. Manning.

הסקה

ההסקה על המודל הבסיסי - העברנו כל משפט מתוך סט ההסקה למודל*, עבור כל משפט קיבלנו מהמודל מטריצת score - אותה העברנו לאלגוריתם ה CLE שהחזיר לנו את הקשתות המתאימות שהוא חזה.

ההסקה על המודל המתקדם מעט מורכבת יותר, תחילה בנינו מילון של אמבדינגס מאומנים מראש* לכל המילים מסט ההסקה. לאחר מכן הזנו את המודל באמבדינגס המאומנים מראש ביחד עם המשפטים עצמם - את הסיבה לכך רשמנו בבולט 5 של השיפורים למודל המתקדם. מכאן בדומה למודל הבסיסי קיבלנו מטריצת score מהמודל שאותה העברנו לאלגוריתם ה CLE.

* מילים לא ידועות קיבלו token של unknown.
* הטענת האמבדינגס של סט ההסקה נעשה **רק** בזמן ההסקה ולא בזמן האימון.

מבחן

על המודל הבסיסי קיבלנו אחוז דיוק UAS של 88.3% על סט המבחן. 0.8832894520322527
עם זמן הסקה של 9.14 שניות.

על המודל המתקדם קיבלנו אחוז דיוק UAS של 91.6% על סט המבחן. 0.9159535955241073

עם זמן הסקה של 27.429 שניות.

ההבדל בביצועים יכול לנבוע ממספר סיבות:

1. הרשת המתקדמת ממדלת את הבעיה בצורה טובה יותר כפי שרשמנו בבולט הראשון של השיפורים ולכן נצפה לדיוק טוב יותר.
2. מילים לא ידועות - ככל הנראה אחת הסיבות העיקריות לשוני. המודל הבסיסי טעה יותר מהמודל המתקדם מכיוון שהוא מכליל פחות ולכן מתקשה עם מילים לא ידועות. על המודל המתקדם הפעלנו רגולריזציה כבדה יותר כמו זריקת מילים וזריקת שכבות ברשת ביחד עם אימון ארוך יותר. ולכן המודל למד להכליל טוב יותר על מילים לא ידועות.
3. היפר-פרמטרים אופטימליים של המודל המתקדם. כאמור את המודל הבסיסי מימשנו עם פרמטרים של המאמר וללא שום אופטימיזציה, להבדיל מהמודל המתקדם.

חלוקת עבודה

עופרי - מודל בסיסי ופונקציות עזר
רוברט - מודל מתקדם ודו"ח מסכם