עיבוד שפה טבעית שיעורי בית 2 דו״ח

יונתן אשלג - 324292143
טל דוגמא - 326614617

אימון

עיבוד הנתונים:
ראשית לכל המודלים השתמשנו במודל Glove-twitter-25 מחבילת Gensim אשר נותן לכל מילה ייצוג וקטורי בגודל 25 המיוצר באמצעות Glove המאומן במיוחד על דאטא של ציוצים מטוויטר. בחרנו להשתמש במודל זה כיוון שהבחנו שהדאטא לקוח מטוויטר.
בתיוגים של המילים בחרנו לפשט את מימד תיוגים לתייג הכל 1 אם זה כל סוג של ישות ו0 אחרת.
במהלך עיבוד הנתונים מצאנו מספר דפוסים חוזרים שלא מקבלים embedding כמו שצריך (מילים oov):

- ״@״: ראינו שהרבה משפטים יש בהם ״@״ ואז שם משתמש כלשהוא שהמודל שלנו לא מכיר אז הוא מקבל "unknown" כדי לפתור את בעיה זו בכל מקרה כזה החלפנו את המילה ב״@@״.
- ״#״: הורדנו את ההאשטאג ממילים שהתחילו בו כדי שהמודל שלנו יזהה אותן.
- לינקים: החלפנו את כל הלינקים ב ״http״
- פיצול מילים: הבחנו שהרבה מילים המכילות בתוכם יותר ממילה אחת (ללא רווח ביניהן) נשלחות לוקטור oov. לכן, יצרנו פונקציה רקורסיבית אשר מחלקת מילים כאלו לתתי-מילים ומחברת בין הוקטורים של תתי המילים. כך, במקרים כמו ״starwars" לדוגמה, הוקטור יהיה $glove("star") + glove("wars")$ אשר אמור להיות דומה למציאות עקב תכונת המשמעות הסמנטית של glove.

טיפול במילים שלא זיהינו לאחר כל השלבים האלה: בהתחלה נתנו לכולם "<unknown>", לאחר מכן גילינו שלהוסיף למילים שמתחילות באות גדולה וקטור קבוע בנוסף לוקטור של "<unknown>״ עבד יותר טוב.
כל השינויים שפורטו כאן שיפרו ביצועים על קובץ הDEV.

<u>תיאור המודלים:</u>
<u>מודל ראשון:</u>
למודל הפשוט בחרנו ליישם לKNN כאשר המטריקה היא מרחק אוקלידי. מספר השכנים שלפיו מסווגים הוגדר להיות 11.

<u>מודל שני:</u>
מודל רשת לינארית: הרשת מקבלת בתור קלט את המילה, צמד המילים שלפניה ואחריה במשפט.
מבנה הרשת:

input_layer= 75 -> 16->8->4->1->sigmoid-> output

כל <- הוא fully connected layer כאשר שמנו בסוף סיגמואיד בשביל לקבל הסתברויות.

פונקציית ה loss שלנו היא binary cross entropy, בה השתמשנו לכל המודלים, מאחר ואנחנו מממשים בכולם תיוג בינארי.
אלגוריתם האיפטום שלנו היה Adam בו השתמשנו לכל המודלים שלנו.
קצב הלמידה שלנו היה 0.0005
המודל שלנו מוציא וקטור בגודל 1 עם ערך בין 0 ל1,הגדרנו סף=0.3 כאשר כל ערך מעל הסף ממופה ל1 (ישות) וכל ערך מתחת מקבל 0 (אין ישות)

<u>מודל שלישי:</u>
מודל LSTM:
אימנו רשת LSTM בעלת 3 שכבות כאשר אחרי השכבה האחרונה יש שכבה אחת fully connected אשר מוציאה מהמרחב הנסתר פרדיקציה בודדת.
קצב הלמידה שלנו היה 0.0005
הסף שלנו היה 0.15
גם פה המודל קיבל והוציא רק 1=ישות 0=לא ישות, התאמצנו רבות לגרום למודל לעבוד בקונסטלציה שבה הוא מקבל את סוגי הישויות השונות, בהתחלה ניסינו לתת לו את כולם, אחר כך הפרדנו ל I,B,0 שניהם לא הצליחו להגיע לביצועים יותר טובים מהמודל שקיבל את התיוגים הבינארים.

<u>מודל רביעי (תחרות):</u>
מודל: LSTM+KNN.
מוטיבציה: כאשר ניתחנו את ביצועי ה KNN הבחנו בתכונה מעניינת ה recall שלו היה ממש גבוה,83.6 ומנגד ה precision שלו היה נמוך,47.2, כלומר: הוא לא מזהה את רוב הישויות אך כאשר הוא מתייג מילה בתור ישות הוא צודק באחוזים טובים מאשר ה LSTM שלנו.
לכן החלטנו לבנות ensemble:
בהזמן הסקה ניתן גם ל LSTM שלנו וגם ל KNN להוציא פרדיקציות. אם LSTM תייג מילה כ0 וKNN כ1 אזי נתייג אותה כ1,בכל מקרה אחר נקשיב ל LSTM.
פרמטרים:
ל KNN הגברנו את K ל11 על מנת להעצים את האפקט שראינו
LSTM: העלנו את הסף מ 0.15 ל 0.20, על מנת להעלות את ה precision, כי ה recall נתמך ע״י ה KNN.

מבחן:

תוצאות:

מודל ראשון KNN :

```
F1 score on train data: 0.6695842450765864
F1 score on dev data: 0.596562184024267
Precision: 0.8082191780821918
Recall: 0.47275641025641024
```

מודל רשת לינארית:

```
F1 score on train data: 0.5418340611353711
F1 score on dev data: 0.5728592889334001
Precision: 0.7636849132176236
Recall: 0.4583333333333333
```

מודל שלישי LSTM:

```
F1 score on train data: 0.5285714285714286
F1 score on dev data: 0.5672227674190383
Precision on train data: 0.6065573770491803
Recall on train data: 0.46835443037974683
```

מודל רביעי LSTM+KNN:

```
F1 score on train data: 0.7141824490386005
F1 score on dev data: 0.6629502572898799
Precision: 0.7130996309963099
Recall: 0.6193910256410257
changed by knn: 107
```

השורה האחרונה סופרת על כמה החלטות ה KNN השפיע.