עיבוד שפות טבעיות

תרגיל בית 1

דו"ח תרגיל בית

<u>מגישים:</u>

- צ'רלי מובאריכי 316278118
- אילן גודיק 316315332

שלב ראשון: השגת פסקאות הטקסט מתוך Ynet

1. הורדנו את דף האינטרנט עם בקשת get לעמוד הכתבה
2. השתמשנו בlxml לשם error-correcting parsing לעץ הxml של הדף
3. השתמשנו בclasses הנמצאים כאב עליון של כל אחד מהרכיבים אותנו רציני להשיג:
    1. `art_header_title` לכותרת
    2. `art_header_sub_title` לתת הכותרת
    3. `art_body` לאב הקדמון המשותף של כל הפסקאות
4. הוצאנו את הטקסט מכל אלמנט p בנפרד
5. ניקינו את הטקסט מartifacts של html:
    1. הורדנו שורות חדשות וtabs, מכיוון שאין להם משמעות בhtml
    2. הורדנו מפרידי פסקאות של html
    3. צמצמנו רווחים רצופים לרווח יחיד
    4. הורדנו שורות ריקות שנוצרו מתהליך זה

בסוף התהליך, קיבלנו פיצול יפה של הטקסט לפסקאות בהתאם למה שרואים בכתבה באתר.

שלב שני: חלוקה למשפטים

האלגוריתם שלנו מזיז חלון בגודל 3 ומחליט בכל הזזה האם הסתיים המשפט במקום זה או לא.
בנוסף, אנו עוקבים על האם אנחנו בתוך מרכאות של ציטוט כרגע, בכדי לסיים משפט בסוף ציטוט.

פונקציית ההחלטה לפיצול שורה:

1. יש לנו אוסף טרמינלים: "?!.;:," שאחריהם נפתח משפט חדש.
2. עבור '.' ו':', אנו פותחים משפט חדש רק אם הם לא מוקפים במספרים, וזאת כדי לטפל במקרים כגון: 3.14 ו15:30.

דילמה: האם לפצל למשפטים בתוך ציטוט?
החלטה: החלטנו כן לפצל למשפטים בתוך ציטוטים, ולאחר ציטוט, גם לפתוח משפט חדש.

אופן הטיפול בציטוטים:

1. לא לפתוח שורה חדשה אם יש טרמינל לפני סוף ציטוט, מאחר ונפתח שורה חדשה לאחר סוף הציטוט.
   לדוגמה: האיש שאל: "מה שלומך?". ואני עניתי: "הכל טוב."
   במקרה זה לא נפתח שורה חדשה לאחר סימן השאלה בציטוט הראשון.
2. אנו פותחים משפט חדש לאחר סגירת מרכאות רק אם יש טרמינל או פסיק לאחר המרכאות, וזאת כדי לטפל בביטויים ששמים במרכאות גם כן.
   לדוגמה:
   - 'בתדרוך עיתונאים הוצגו "מעגלי ההשפעה" של העבודות.'
     לא נרצה לפתוח משפט חדש לאחר סגירת המרכאות במקרה זה.
   - 'יורם אוחיון, סגן מפקד מחוז תל אביב, הדגיש מסר אחד: "רבותיי, תעברו לתחבורה הציבורית". הוא ציין: ...'
     במקרה זה, שישנה נקודה לאחר סוף המרכאות, נפתח שורה חדשה, מכיוון שיותר סביר שזהו משפט שלם בתור ציטוט.

3. תמיכה בקיצורים כמו 'ק"מ': כדי להחליט האם אלו מרכאות פותחות, סוגרות או שזה לא ציטוט בכלל: נבדוק האם המרכאות מוקפות באותיות: אם כן, אז אלה הן לא מרכאות פותחות או סוגרות.

שלב שלישי: Tokenization

לשלב זה, השתמשנו בחלון זז בגודל 4, ופונקציית החלטה של האם לפצל את החלון באמצע, בין 2 התווים הראשונים ו2 התווים האחרונים.

יש לנו תווים שתמיד צריך לפצל אותם לtoken נפרד, והם:
" !?, () ;/\+="
בנוסף, ישנה לוגיקה מיוחדת למרכאות, מקף, נקודתיים, נקודה וכוכבית.

פונקציית ההחלטה לפיצול token:

1. אם אחד התווים באמצע החלון הוא תו שתמיד מפצלים לפיו, פצל.
2. אם אחד התווים האמצעיים הוא מרכאות או מקף, והוא לא מוקף במספרים ואותיות משני צדדיו, פצל. זאת בכדי לא לפצל צירופים כמו "בית-הספר", "ה-15" וקיצורים כמו 'ק"מ'.
3. אם יש נקודתיים שלא מוקפות במספרים, פצל. זאת כדי לא לפצל שעות כגון: "13:30"
4. אם יש כוכבית, שאין אחריה מספרים, פצל. זאת בכדי לא לפצל מספי טלפון כגון "*123"
5. אחרת, אל תפצל לtoken במיקום זה.

בעקרון הבדיקות הן על חלון של 3, אך עושים זאת פעמיים בקבלת גודל חלון של 4, וזאת בכדי לפצל גם לפני וגם אחרי כל מקום פיצול token מתאים.

בנוסף, אנחנו ניקח נקודות רצופות בתור token יחיד, כגון: "אני לא בטוח..." יהפוך למילים כtokens וגם שלושת הנקודות יחדיו כtoken יחיד.
בנוסף, אנו לא מפצלים את $, %, ₪ מכיוון שהם מחזיקים את המשמעות של המספר הצמוד להם, ואם מפצלים, מאבדים משמעות זו.
גם @ ו_ החלטנו לא לפצל לtoken נפרד, מכיוון שבד"כ משתמשים בהם בכתובות אימייל, ולא נרצה לחתוך כתובת אימייל באמצע.
בנוסף, לא פיצלנו גרש יחיד ', וזאת מכיוון שבד"כ שימושו הוא שינוי אות, כגון ג', וקיצורים כגון וכו', וגרש זה מספק את משמעות המילה.

הערות על הקוד

כל הקוד מתועד היטב, עם כל ההסברים שנתנו בדו"ח זה, בנוסף להסברי מימוש.

בנוסף, צירפנו את כל הunit tests שלנו בתור הערות בקוד הסופי.