תרגיל 4
Word Embeddings

מבוא

בתרגיל הזה נתנסה בכמה שימושים שיש ל-word embeddings: משחקי דמיון מילים, מודלי שפה ושימושים נוספים. להתרשמות נוספת מ-word embeddings, אם אתן מכירות את המשחק Wordle, וודאי תשמחו לשמוע על גרסת ה"חם-קר" הסמנטית שלו – Semantle – שבבסיסה שימוש בוקטורי מילים! נעבוד עם קבצי word embeddings של GloVe, שבהם ייצוג של 400,000 מילים באנגלית שאומנו על טקסטים מויקיפדיה ו-Gigaword. מפאת גודלם של הקבצים, הם לא מצורפים לאתר, לכן אתן מתבקשות להוריד את קבצי הטקסט מכאן. הורידו את הקובץ glove.6B.zip, שבו 4 קבצי טקסט המכילים את וקטורי המילים ב-4 אורכים שונים. נשתמש בייצוגים ב-50 מימדים לצורך התרגיל. יש להמיר את קובץ הטקסט לקובץ word2vec ולשמור אותם באופן הבא (נמצא מושחר בקובץ ה-template המצורף לתרגיל):

```
## Do the following only once!

## Save the GloVe text file to a word2vec file for your use:
# glove2word2vec(<downloaded_text_filename>, <full_path_vector_filename>')
## Load the file as KeyVectors:
# pre_trained_model = KeyedVectors.load_word2vec_format(<full_path_vector_filename.kv>, binary=False)
## Save the key vectors for your use:
# pre_trained_model.save(<full_path_keyvector_filename.kv>)

## Now, when handing the project, the KeyVector filename will be given as an argument.
## You can load it as follwing:
# pre_trained_model = KeyedVectors.load(<full_path_keyvector_filename>.kv, mmap='r')
```

הפקודה הראשונה אחראית לייצר קבצי Word2Vec. שמרו קובץ זה (ע"י הפקודה) בנתיב לשימושכן. אחרי כן, טענו את הוקטורים כ-KeyVectors, ושמרו אותם בקובץ עם סיומת kv. אחרי שעשיתן זאת, השחירו את קטע הקוד הנ"ל (כלומר, יש לשמור את הקבצים באופן חד-פעמי). שם קובץ ה-kv יינתן כקלט לתכנית, אותו ניתן לטעון כפי שכתוב בשורה האחרונה של קטע קוד זה. קטע הקוד נמצא בקובץ ה-template המצורף לתרגיל. בתרגיל זה אין דרישה למימוש של מחלקה ייעודית.

חלק א': התנסות ראשונית

לפני שנגש לתרגיל, נרצה להתנסות מעט עם וקטורי המילים, ובעיקר עם תכונה חשובה ביותר שלהם: מרחקים.

1. הכינו רשימה של 10 זוגות מילים שלהשערתכן יש ביניהן קשר כלשהו: הפכים, מילים נרדפות, מילים קרובות במשמעות (לדוגמא apple ו-pear, או love ו-hate). הציגו באמצעות הפונקציה similarity את מרחק הקוסינוס ביניהן. כתבו את הרשימה עם המרחקים בקובץ הפלט באופן הבא:

```
Word Pairs and Distances:
1. <word_1> - <word_2> : <distance>
...
10. ...
```

צרפו את הרשימה לדו״ח, ונסו לתת השערה למרחקים שקיבלתן.

2. חשבו על 5 זוגות של אנלוגיות (כל אנלוגיה מורכבת מזוג של זוגות מילים, לדוגמא, man:woman ו-boy:girl, או hat:head ו-shoe:foot). עבור כל אנלוגיה השמיטו מילה אחת, והשתמשו בפונקציה most_similar על-מנת לנסות לקבלה דרך ביטוי אריתמטי. לדוגמא, על-פי האנלוגיה נגזור את הביטוי girl = boy + woman – man, נבדוק לפי הפונקציה:

```
word = word2vec_model.most_similar(positive=['boy', 'woman'], negative=['man'])
```

השוו את המילה שהתקבלה למילה המקורית בעזרת הפונקציה similarity. כתבו בקובץ הפלט את האנלוגיות, המילה שהתקבלה והמרחק בין המילה שהתקבלה למקורית באופן הבא:

```
Analogies:
1. <word_1> : <word_2> , <word_3> : <word_4>
2. ...
5. ...

Most Similar:
1. <the arithmetic phrase > = <the word the function returned>
2. ...
5. ...

Distances:
1. <expected> - <returned> : <distance>
2. ...
5. ...
```

חלק ב׳: הדרך הקלה ל-Grammy

אחרי שסיימתן את הקורס ״עיבוד שפות טבעיות״ קיבלתן שלל הצעות עבודה מעניינות ומאתגרות, אך הצעה אחת בלטה במיוחד: לעזור למוזיקאים צעירים בתחילת דרכם לכתוב שירים. לאחר שנעניתן להצעה המפוארת, התחלתן מיד במלאכה, אך גיליתם שהמצב של המוזיקאי הראשון איתו תעבדו חמור משחשבתן: הוא אינו יודע

לחרוז, חסר חוש קצב וכתיבתו עילגת. הבנתן שאין ברירה אלא לשאוב השראה מהשיר שזכה בתואר "שיר השנה" ב-Grammy's, ותשתמשו בכל הכישורים שצברתן בקורס הנ"ל.

בנו תכנית Python המאפשרת לשנות מילה מסוימת במשפט באופן הבא:

1. ראשית, מאחר והשיר מגיע כטקסט גולמי (להבדיל מטקסטים מה-BNC), בצעו טוקניזציה למשפט בעזרת המימוש שלכן בתרגיל 1 (במקרה זה אין צורך להתייחס למקרי קצה, מאחר וסימני הפיסוק משמשים רק כסימני פיסוק).
2. עבור המילה הנתונה אותה תרצו לשנות, השתמשו במודל word embedding על-מנת למצוא את 10 המילים הקרובות ביותר אליה.
3. החליפו את המילה הנתונה באחת מהמילים הקרובות ביותר כך:

- אחד הקלטים לתרגיל יהיה תיקיית קבצים, ממנה תיצרו קורפוס (כפי שעשיתן בתרגילים קודמים).
- אם המילה שבחרתן להחליף אינה הטוקן הראשון או האחרון במשפט, השתמשו במודל ה-trigrams מתרגיל 2 ועל הקורפוס שתצרו כדי לבדוק את מספר המופעים של כל trigram בהחלפת כל מילה מ-10 המילים שנבחרו (עבור המילה, הטוקן הקודם והעוקב). לדוגמא, אם נרצה להחליף את המילה apple מתוך המשפט I bought an apple today, לכל מילה $ מעשרת המילים הכי קרובות ל-apple נספור את מספר ה-trigrams 'an $ today', והמילה שתניב את הספירה המרבית תוחלף). אם בחרתן להחליף מילה בתחילת/סוף משפט, התייחסו למספר המופעים של המילה הקודמת/העוקבת כ-0.
- אם לא קיים trigram כנ"ל בקורפוס (מספר כל המופעים הוא 0), החליפו במילה עבורה סך ה-bigrams מירבי (לדוגמא, במשפט הקודם נרצה להחליף את המילה apple במילה $ עבורה סכום מספר מופעי הצירוף $ an והצירוף $ today מירבי).
- אם גם פה מספר המופעים 0, החליפו את המילה הנתונה במילה הקרובה ביותר אליה.

לתרגיל מצורף קובץ טקסט ובו מילות השיר Leave the Door Open מאת Anderson .Paak, Bruno Mar ו-Silk Sonic שזכה בתואר "שיר השנה" ב-Grammy's. כל שורה בשיר כתובה בשורה נפרדת. עבור **כל שורה**, בחרו **לפחות מילה אחת** אותה תרצו לשנות: תרצו שהשיר שלכן יהיה להיט, אך שאף שורה (בעלת תוכן) בשיר לא תהייה זהה לשיר המקורי. הפעילו את הפונקציה לשינוי המילה כדי לעשות זאת (כלומר, בצעו טוקניזציה, והפעילו את הפונקציה). בחירת המילים צריכה להיעשות ידנית (על-ידכן). הדפיסו את השיר החדש לקובץ הפלט באופן הבא:

```
=== New Hit ===

< you new hit song, line by line >
```

צרפו לדו"ח את השיר החדש שיצרתן. כמו כן, ענו בדו"ח:

1. האם התוצאות שהתקבלו יצרו משפטים הגיוניים וקוהרנטיים?

2. אם לא, אילו מילים לא הוחלפו במילים מתאימות? כיצד (תחבירית או סמנטית)?

חלק ג׳: מפת הציוצים

עד כה ראינו איך ניתן להשתמש בוקטור כמייצג מילה אחת, אך ניתן גם ליצור וקטור המייצג קבוצת מילים (משפט, פסקה וכד׳) ע״י ממוצע משוקלל של וקטורי המילים בקבוצה. כך נקבל ייצוג וקטורי של טקסט לפי תוכן. בחלק זה נתנסה בייצוג וקטורי של ציוצים (tweets) מ-Twitter, ונציג אותם גרפית.

- מצורף קובץ טקסט ובו ציוצים מהשנה האחרונה ב-4 נושאים: Covid, האולימפיאדה וחיות מחמד. הציוצים ממוספרים, וכתובים בקובץ בפורמט הבא:

```
=== Tweet Category 1 ===
1. < First tweet of category 1 >
2. < Second tweet of category 1 >
. . .
=== Tweet Category 2 ===
8. < First tweet of category 2 >
9. < Second tweet of category 2 >
```

כשבין ציוץ לציוץ באותה קטגוריה יש ירידת שורה אחת, ובין קטגוריות ישנן 2 שורות.
מטרת הסעיף הזה היא לבדוק האם ייצוג של טקסט באמצעות וקטורי מילים יכול לסייע בסיווג טקסטים, במקרה זה, סיווג לפי קטיגוריות של הציוצים.

- כמו בסעיף הקודם, גם כאן הציוצים נתונים כטקסט גולמי (שלא עבר טוקניזציה). בצעו טוקניזציה לציוצים, באותו אופן כמו בחלק הקודם.
- ננסה כמה ממוצעים משוקללים לוקטורי הציוצים. על מנת לחשב את אלו, ניתן למילים בציוץ משקלים שונים: בנו 3 פונקציות משקל לכל מילה, כאשר הפונקציה יכולה לקבל כקלט מילה (טקסט), וקטור או כל מימוש אחר שתבחרו: האחת מייצגת ממוצע אריתמטי, לכן תחזיר 1 לכל קלט; השנייה תחזיר משקל (ממשי) רנדומלי לכל מילה בטווח בין 0 ל-10 (לא כולל); השלישית היא פונקציית משקל לבחירתכן. פרטו בדו״ח איזו פונקציה בניתן ואת השיקולים שלכן. את הוקטורים חשבו באופן הבא, עבור כל פונקציית משקל:

$$\left(\sum_{i=1}^{k} w_i * v_i\right) \Big/ k$$

כאשר w_i מייצגת את משקל המילה ה-i, v_i הוא וקטור המילה ה-i ו-k הוא מספר המילים בציוץ.

- כפי שלמדת במבוא ללמידה ממוכנת, PCA היא שיטה לינארית להורדת מימד ע״י הטלה בכיוונים ה״משפיעים״ ביותר (פירוט על השיטה למעוניינות כאן). השתמשו ב-PCA הממומש בספריית sklearn כדי להוריד את מימד כל וקטור ל-2. הציגו באמצעות pyplot את הוקטורים המייצגים כל ציוץ, באמצעות גרף נפרד לכל פונקציית משקל. בכותרת הגרף רשמו את שם פונקציית המשקל ואת שמכן, וליד כל נקודה המייצגת ציוץ רשמו את הקטגוריה של הציוץ ואת מספר הציוץ (לפי הקובץ).

צרפו את הגרפים לדו"ח, וענו:

1. באילו גרפים (כלומר, ע"פ איזו פונקציית משקל) ישנה הפרדה מובהקת בין הקטגוריות, אם בכלל?
2. אילו ציוצים הופרדו הכי טוב באופן קונסיסטנטי? נסו לתת השערה מדוע, על סמך הסתכלות בציוצים.
3. מה ניתן להסיק מכך על היכולת לבנות מסווג ע"פ וקטורי מילים?

אופן הגשה

1. לתרגיל מצורף קובץ Python בשם **<surname>_<first_name>_ex4.py**. עליכן לממש את התכנית (בגרסה 3.5 ומעלה) ולשנות את שם הקובץ בהתאם (לדוגמא: zeira_yuli_ex4.py). **אין להשתמש בספריות nltk, spacy או transformers**. על הקובץ לרוץ תחת הפקודה:
   python <name_ex4.py> <kv_file> <xml_dir> <lyrics_file> <tweets_file> <output_file>
   (כמצוין בקובץ עצמו). **לקובץ הפלט יש לכתוב בקידוד utf-8**. הקפידו שזמן הריצה של התכנית לא יעלה על 5 דקות.
2. קובץ PDF בשם **<surname>_<first_name>_report4.pdf** (לדוגמא: zeira_yuli_report4.pdf) ובו דו"ח המפרט על הקוד שכתבתן ועל ההחלטות שקיבלתן במהלך העבודה. על הדו"ח להיות באורך שני עמודים לכל היותר.

*** יש להגיש את שני הקבצים הללו בנפרד, ולא כקובץ zip (ודומיו) ***

יש להקפיד על עבודה עצמית, צוות הקורס יתייחס בחומרה להעתקות או שיתופי קוד.
ניתן לשאול שאלות על התרגיל בפורום הייעודי לכך במודל.
יש להגיש את התרגיל עד לתאריך 14.5.22 בשעה 23:59.

בהצלחה!