**פרויקט תכנות – בניית מנוע לאחזור מסמכים**

1. פירוט גישות מימוש:
2. search\_engine\_1 - שיטת word2vec:

בשלב הranker עבור כל מסמך רלוונטי ביצענו חישוב של ווקטור ממוצע של המסמך על פי מודל word2vec. מצאנו את הווקטור של כל מילה במודל וחישבנו ממוצע של כל ווקטורי המילים. עבור כל ווקטור ממוצע ביצענו חישוב cosine similarityבין ווקטור השאילתה לווקטור המודל. כדי להשתמש במודל word2vec ללא אימון שלו על הדאטה הספציפי שלנו השתמשנו במודל מאומן של גוגל.

1. search\_engine\_2 - שיטת gloVe:

בשלב הranker עבור כל מסמך רלוונטי ביצענו חישוב של ווקטור ממוצע של המסמך לפי מודל gloVe וביצענו המרה של הוקטור המתקבל לוקטור מפורמט word2vec. מצאנו את הווקטור של כל מילה במודל וחישבנו ממוצע של כל ווקטורי המילים. עבור כל ווקטור ממוצע ביצענו חישוב cosine similarityבין ווקטור השאילתה לווקטור המודל. השתמשנו במודל מאומן של שמתמחה בציוצים ובחרנו להשתמש במודל שמפיק וקטורים באורך 25 כדי שהחישוב יהיה מהיר.

1. search\_engine\_3 - שיטת Advanced Parser:

בשלב הParder בחרנו לממש חוקים שונים כדי שהמילון יהיה מצומצם ככל הניתן ע"י סינון terms שלא מועילים או לא רלוונטיים.

החוקים הם:

* + השארת תווים שהם Basic Latin בלבד.
  + הורדה של הסימנים הבאים: .,!?()~:; מ-full\_text
  + הורדה של n\ ו- t\ (סימון ירידת שורה או טאב) מ-full\_text.
  + retweet
  + חוקי url:
    - הפרדת מילים שמופרדות עם מקף כך שלא נשמור את ה(-).
    - הפרדת מילים שמופרדות ע"י: \//\:/?=\-&+ למניעת שמירה של סימנים אלו וכן שמירה של תוכן רלוונטי מתוך הנתיב.
    - שמירת המילים status וweb רק אם לא הגיעו מתוך url.
  + הפיכת המילים dollars וdollar לסימן $.
* שילוב חוק המספרים והדולר - החלפה של המילים dollars, dollars בסימן $. מאחד שלושה terms שונים תחת term יחיד.
  + לדוגמא tweet\_id מספר 1288845008884436995:

10,000 dollars🡪10K$

* + הרחבת הstop words: בחרנו להוריד מהמילון מילים אשר מופיעים בקביעות אך לא תורמים לאחזור.

המילים הם: 'rt', 'www', 'http', 'https', 'tco', 'didnt', 'dont' 'twitter.com'

* + הורדה של כל הterms שמופיעים פעם אחת במילון:

בחרנו להוריד terms אלו מכיוון שהם לא משמעותיים, הביטויים המשמעותיים והרלונטיים לקורפוס מופיעים יותר מפעם אחת. לרוב הם לא יופיעו במודלים המאומנים שבהם אנו משתמשים ולכן כך אנחנו מקטינים את המילון ומאפשרים אחזור מהיר ויעיל יותר.

1. search\_engine\_4 - שיטת wordnet:

לקסיקון לשפה האנגלית המכיל מושגים, כשהמשמעויות השונות של כל מילה שמופיעה בו מפורטות. לכל מושג (משמעות) מצורפת רשימת המלים הנרדפות שלו וכן מלים שקשורות אליו בקשרי משמעות שונים, כגון, קשרים היררכיים של כלל ופרט ושלם מול חלקיו.

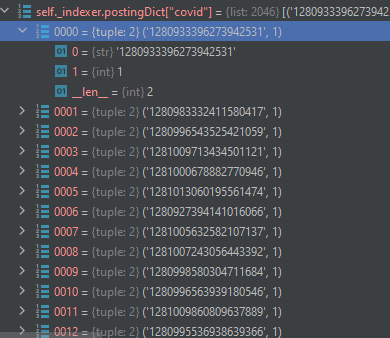
השתמשנו בשלב הsearcher

1. search\_engine\_best – שילוב של word2vec ו Advanced Parser:
2. יכולות שנוספו וירדו בחלק ג':
3. כמות מילים במילון: 4514
4. מידע על כניסה באינדקס

דוגמא למבנה אינדקס ופוסטינג של מילה:

self.\_indexer.inverted\_idx["covid"] = {int} 2046

self.\_indexer.postingDict["covid"] = {list:2046}



1. פעולות להגעה לזמן ריצה מיטבי:
2. פעולות לשיפור התוצאות:
3. בחירת שיטה מיטבית:

תחילה הרצנו את כל אחת מהשיטות בנפרד כדי למדוד את התוצאות ולקבל מושג על יעילות כל אחת מהשיטות, לאחר מכן, בחרנו להשתמש ב Advanced Parser מכיוון ששיטה זו יוצרת לנו מילון מצומצם ויעיל ככל הניתן. החלטנו לנסות לבצע שילוב של שיטה זו יחד עם השיטות האחרות שמימשנו ולמצוא את השילוב שיתן לנו את התוצאות הטובות ביותר בכל הפרמטרים הנבדקים.

וכך גילינו כי שילוב השיטה Advanced Parser יחד עם \_\_\_\_\_\_\_ מפיק לנו זמן אחזור עבור כל שאילתא מהיר וכן precision וrecall בעלי תוצאות טובות ביותר.

1. שימוש בקוד פתוח:

* word2vec - על מנת להשתמש במודל word2vec ללא אימון שלו על הדאטה הספציפי שלנו השתמשנו במודל מאומן של גוגל.

קישור למקור: <https://machinelearningmastery.com/develop-word-embeddings-python-gensim/>

* gloVe – השתמשנו במודל מאומן של אוניברסיטת סטנפורד.

קישור למקור: <http://nlp.stanford.edu/data/glove.twitter.27B.zip>

* wordnet -

1. מידע נוסף:
2. הסבר על חזרת ציוצים:
3. מדדי ביצועים:
4. מנוע best:

יתרונות:

1. ג
2. ג
3. ג

חסרונות:

1. ג
2. ג
3. ג