דוח מנוע חיפוש חלק ב'

1. פירוט השיטות:

1. Advanced Parser:

בשיטה זו החלטנו לשפר את הביצועים של הפרסר על מנת לתפוס את המילים החשובות ביותר והרלוונטיות ביותר של המסמך. שלב הParser- מבצע בדיקה ועמידה בחוקים הדרושים, פירוק הטקסט לterms כנדרש כאשר ביצוע הפעולות הנ"ל מתבצעות בעת קריאה לפונקציה המתאימה במחלקה, המחלקה יוצרת ומחזירה אובייקט מסוג document לכל מסמך שעבר תהליך parse.

המחלקה מתחלקת לשלושה חלקים עיקריים –

* ביצוע Parse למסמך כולו, הפונקציה הראשית שאחראית לקריאת לפונקציות המבצעות את הParse לחלקים ספציפיים במסמך. *(Parse\_doc())*
* ביצוע Parse למשפט יחיד.*(Parse\_setence())*
* ביצוע Parse לurl.*(Parse\_url())*

נתונים שהרצת הפרסר יוצרת שעוזרים לנו בהמשך:

*Dict\_upper\_word* – מילון שמכיל את כל המילים שהם באותיות גדולות וכמות הפעמים שהופיעו בטקסט.

*Entity\_dict* – מילון המכיל את כל הישויות העשויות להיות בטקסט וכמות הפעמים שהופיעו בטקסט.

*Retweet\_dict* – מילון שמכיל את כל הציוצים שעשו להם שיתוף.

Tweet\_Date - שומר את התאריך של יצירת הציוץ.

שדות נוספים:

*Stop\_words* – מאגר מילים בנוי של הספרייה NLTK מילים אלו נחשבות כמילים לא משמעויות ואותם נרצה להוריד ממילון המונחים הסופיים. בחלק זה החלטנו להשאיר ולהשתמש במאגר זה על מנת שנוכל לקבל דיוק גבוה יותר של מונחים רלוונטים לכל ציוץ.

*Stemmer\_status* – משתנה בוליאני שמבטא האם נרצה לבצע stemmer על המילה.

*url* – במידה וקיים url בטקסט נשמור אותו ונבצע עליו parse בנפרד.

נסביר על שתי שיטות עיקריות במחלקה ועל הכללים והחוקים התבצעים בהם:

*Parse\_setence(text)* – השיטה מקבלת את כל הטקסט שעליה לבצע parse.

הפונקציה מחזירה את הנתונים הבאים:

*Final\_term* – מילון מונחים סופי לאחר שעברנו על כל המילים בטקסט וביצענו עליהם את כלל החוקים וכמות הפעמים שהופיע המילה בטקסט.

*Entity\_dict* – מילון מונחים סופיים של כלל המונחים העשויים להיות ישויות וכמות הפעמים שהופיעו בטקסט.

אופן הפעולה של הפונקציה:

הפונקציה מבצעת מחיקה של התווים הבאים [...,\n,-,,] ואז מבצעת split לפי רווחים ומחזירה מערך עם המחרוזות המופרדות.

לאחר מכן הפונקציה עוברת בלולאה על כל המחרוזות במערך ומטפלת בכל אחת לפי חלוקה למקרים וביצוע מספר בדיקות:

תחילה נבדוק האם המילה מכילה אותיות ותווים מיוחדים שעלולים לגרום לנו לבעיות בהמשך התהליך.

לאחר מכן נוודא המילה אינה ריקה ונתחיל בתהליך החוקים.

1. אם המילה מכילה את הסימן '-', '/', '\_' :

נבצע הפרדה של המילים לפי התו הנ"ל ונכניס אותן לסוף רשימת המילים המיועדות לבדיקה.

2. אם המילה מכילה את הסימן '%':

נשלח את המילה לפונקציית עזר שתבדוק האם זהו מספר שהוא אחוז ותנקה אותו מסימנים מיותרים ותכניס אותו לרשימת המילים הסופיות.

3. אם המילה מתחילה או מסתיימת עם הסימן $:

נשלח לפונקציית עזר שתוריד סימנים מיותרים ותבדוק האם זה באמת מספר, על מנת שנוכל להתייחס למונח זה כסכום כסף.

4. האם המילה היא מספר:

בפונקציה זו אנחנו מטפלים בכמה אפשרויות:

* מספר רגיל שאין לו יחידות – נכניס אותו לפונקציית עזר הבודקת את היחידות המתאימות לפי גודל המספר ומבצעת את ההתאמות לפי שכתוב בהנחיות העבודה.

במידה ונקבל 1000000 נחזיר 1M.

* מספר עם מילת יחידות אחריו – נבדוק האם המילה שמגיעה אחרי המספר היא מילת יחידת ונחזיר אותם מחוברים לדוגמא: 1milliom נחזיר 1M.
* מילה עם מילה שמסמנת לנו אחוז- נבדוק אם המילה היא חלק ממילים שיכולות להיות אחוז ונחזיר את המספר עם הסימון %.

5. האם המילה היא url – נבדוק האם המילה מתחילה עם http ובמידה וכן נשמור אותה על מנת לבצע עליה parse בהתאם בחוקים של url.

במידה והמונח לא נכנס לאף קטגוריה למעלה נכניס אותו לסט בדיקות של מונחים שבטוח מכילים מילים.

תחילה נבצע נקיון על המילה שמכילה סימנים שמוגדרים ברשימת סימנים למחיקה.

6. האם המילה מתחילה ב@:

נשמור את המונח כמו שהוא ונכניס אותו אל הרשימה.

7. האם המילה מתחילה ב#:

נשלח לפונקציה עזר שמחלקת את ההאשטאג למילים לפי מילים אותיות קטנות גדולות, לפי סימנים מיוחדים, ולפי הכתוב בהוראות במידה וקיים '\_', בנוסף, לכל מילה שהפרדנו נהפוך את כולה לאותיות קטנות ונבצע עליה stemmer במידת הצורך.

לאחר מכן נבצע הפרדה האם המילה מורכבת ממספר אותיות או שמכילה רק שתי אותיות, במידה והיא אות יחידה אנחנו נסיר אותה מכיוון שזה יכול להיות טעות או מילים שפחות יהוו לנו משמעות בשאילתה.

בסט בדיקות זה לפני שנכניס את המילה למילון נבדוק שהיא אינה מופיעה ברשימת המילים stop words ונבצע עליה stemmer במידת הצורך.

8. נבדוק האות ההתחלתית היא אות גדולה:

נכניס את המילה למילון בצורה כך שכל המילה תהיה באותיות גדולות. (את הבדיקה של אותיות קטנות/גדולות נבצע ב-indexer על מנת לחסוך זמן ריצה וכדי שנוכל לבדוק את המילים מול הקורפוס כולו)

9. בדיקת ישויות:

כאשר מתגלה מילה עם אות גדולה נכניס אותה לרשימת ישויות אופציונלית, ונשמור את מיקום המילה במסמך. כאשר נמצא מילה נוספת שהיא עם אות גדולה נבדוק האם המיקומים עוקבים. במידה וכן, נתייחס למונח כישות ונכניס אותו לרשימת הישויות האופציונלית גם כן.

במידה והמילה העוקבת אינה אות גדולה, נבדוק כמה מילים יש לנו ברשימת הישויות האופציונלית. במידה וזה יותר ממילה אחת נחבר בין המילים לישות אחת ונכניס אותה למילות ישויות של המסמך שבו נספור גם את כמות המופעים של הישויות במסמך.

במידה וזו מילה יחידה עם אות גדולה לא נתייחס אליה כישות.

(בדומה לאותיות גדולות קטנות נבצע בדיקת ישויות נוספות מול הקורפוס כולו בשלב הindexer).

כל מילה אחרת תיכנס למילון כמילה באותיות קטנות.

את שלב זה נבצע על שני ערכים שקיימים לנו בנתונים על המידע:

1. הציוץ המלא – כל התוכן המילולי שקיבלנו מהציוץ.

2. Retweet – במידה והציוץ משותף נבדוק האם יש טקסט שנוסף לשיתוף ונבצע עליו את שלב הפרסור על מנת שנוכל להרחיב את המידע על תוכן הציוץ.

Parse\_url:

תחילה נבצע בדיקה האם קיים לנו url בתא המתאים שקיבלנו מכל המידע שקראנו על

המסמך. במידה וקיים לנו url מלא, נבצע עליו את ריצת הparse. אחרת, אם קיים לנו נתיב

מקוצר נשלח אותו. במידה ולא קיים לנו נתיב בכלל לא נבצע תהליך parse זה למסמך.

url\_handler(val,tweet\_id) – פונקציה זו מקבלת נתיב ואת המספר הייחודי של הציוץ.

תחילה היא מבצעת split לפי הסימנים [/@!#$)(,~?%\*&:;]. לאחר מכן נעבור על כל מונח שהתקבל מהפירוק ונכניס אותו לעץ אפשריות על מנת לבדוק אותו לפי הכללים והחוקים שנגדיר.

1. במידה המילה מכילה '-':

נבצע split לפי התו הנ"ל ונכניס כל אחד מהמילים שהתקבלו מהפירוק את רשימת המילים הממתינים לבדיקה.

2. נבדוק שאורכו של המונח הוא גדול מתו אחד:

בדומה בריצת הparse שנעשה על הטקסט מילים שהם בגודל של תו אחד אינם מביאים לנו משמעות במסמך.

3. במידה והמונח מתחיל ב'www':

נפריד את המונח לשני חלקים ונשמור רק את החלק של הדומיין. את המונח www החלטנו כי אין לנו צורך לשמור מכיוון שהוא לא מהווה לנו מידע רלוונטי בעת החיפוש.

4. במידה והמונח שווה למספר הייחודי של הציוץ:

החלטנו כי לשמור את המספר הייחודי של הציוץ אינו מהווה לנו מידע שימושי, והנתיב היא רק קישור אל הציוץ עצמו ברשת ולכן החלטנו שלא להוסיף אותו לרשימת המונחים שלנו.

5. רשימת stop words:

החלטנו שעל מנת לייעל את תהליך הparse לא נוסיף מילים אל רשימת הstop word הקיימת אלא נבנה אחת משלנו שתכיל את כל המילים שאנחנו לא רוצים לשמור ברשימת המונחים הסופית.

נבדוק האם המילה לא נמצאת ברשימה הנ"ל ונוסיף אותה לרשימת המונחים הסופית של המסמך.

פונקציות נוספות שמהוות חשיבות רבה במחלקה זו:

Popularity\_retweet(url\_list,id\_tweet) – ציוצים רבים עושים שיתוף לציוצים אחרים וראינו שניתן להשתמש במידע זה ולקבל ממנו ערך חשוב שמייצג את פופולריות המסמך.

הגדרת הפופולריות – ככל שיותר מסמכים עשו למסמך i שיתוף ככה הפופולריות של מסמך i עולה.

פונקציה זו עוברת על המידע שקיבלנו על המסמך ובודקת האם קיימים הנתיבים שמייצגים את הציוצים להם הם עשו שיתוף.

ניקח את הנתיבים הנ"ל ונחלץ מהם את המספר הייחודי של הציוץ שאותו שיתפו. נשמור את רשימת הציוצים ובהמשך בתהליך הindexer נבצע בדיקה כמה פופולרי כל מסמך.

Tweet\_date\_parse(tweet\_date) – ראינו לנכון לשמור את תאריך הפרסום של הציוץ על מנת שיוכל לעזור לנו בתהליכים מתקדמים. לכן לקחנו את המידע שקיבלנו כאשר קראנו את המסמך, בודדנו את תאריך הפרסום ורשמנו אותו בתבנית בה יהיה לנו נוח להשתמש. (xx/xx/xxxx)

1. WordNet-

שיטה זו מבצעת הרחבה של השאילתה ומוסיפה מילים בעלות משמעות דומה למילים בנמצאות בשאילתה. שיטה זו עוזרת לנו לאפשר למצוא מסמכים שיהיו רלוונטים בצורה הגבוה ביותר בתוכנם ובמשמעותם.

על מנת לממש שיטה זו יצרנו מחלקת WordNet שתהיה אחראית על כך, השתמשנו בספריית WordNet של NLTK.

אופן הפעולה של השיטה:

השיטה מקבלת רשימה של term שמייצגים את השאילתה לאחר ביצוע Pasre.

נעבור על כל מילה ברשימה ונבצע עליה את הפונקציה Synsets שמביאה לנו משמעויות שונות של המילה אותה הכנסו. לפי בדיקה מקיפה של המילים המוחזרות ראינו כי המילים הראשונות המוחזרות הן המילים שמייצגות את המונח בצורה הטובה ביותר והכי דומות למילה שהוכנסה.

בנוסף, ערכנו מספר בדיקות לגבי כמות המילים שניקח מכל מילה שנבצע עליה הרחבה והוחלט כי המספר שעוזר לקבל את השאילתות הרלוונטיות ביותר הוא 2.

לאחר שנרחיב את השאילתה, נמשיך בביצוע החיפוש והדירוג הרגיל שלנו.

2. שינויים והוספות ל-Parser:

הוספות –

תוכן מציוצים משותפים- ראינו כי במידה והציוץ ביצע שיתוף ניראה כי קיים תוכן נוסף במידע שאנחנו מקבלים. לכן בדקנו האם קיים תוכן מילולי שמייצג לנו את הציוץ ששותף ולאחר ביצוע פרסור הוספנו את ה-terms שהתקבלו לרשימת המילים שמוחזרת. ראינו כי הוספה זו מאפשר לנו לקבל קשת רחבה יותר של מילים שמתארות את המסמך וכך נוכל לדייק במשמעות תוכן המסמכים לשאילתה.

מילה בעלת 2 אותיות- ראינו כי הרבה מילים בעלות 2 מילים אינן מספקות לנו מידע רלוונטי על המסמך ולכן החלטנו כי מילים בעלות 2 אותיות לא נכניס למילון שלנו על מנת שהמילון יכיל את המילים המשמעותיות ביותר.

הורדות-

Stemmer- החלטנו כי לא נבצע Stemmer על הביטויים המתקבלים. פעולה זו לוקחת הרבה זמן ומאטה את קצב הביצועים ובנוסף, ראינו כי לפי המדדים שנבדקו המסמכים המוחזרים לשאילתה לאחר ביצוע ה-Stemmer פחות רלוונטיים לשאילתה מאשר אם כן נבצע Stemmer.

2. כמות מילים: