עיבוד שפות טבעיות - תרגיל 2 - מודלי שפה

N-GRAMS

שלב 1 - בניית מודלי שפה

השתמשנו בקובץ האקסל שבמודל

בשלב זה כנדרש בנינו שני מודלים בהתאם.

בנינו מחלקה בשם Trigram_LM שמתאימה לכל אחד מהמודלים.

מחלקה זו כוללת בתוכה את הפונקציות הבאות:

__init__:

פונקציה זו מקבלת ארגומנט protocol_type

שמציין את סוג הקורפוס ,היא קוראת את קובץ האקסל וטוענת את הדאטה המתאימה לפי סוג הקורפוס.

pre_process:

פונקציה זו מעבדת את המשפטים בקורפוס , סופרת את מספר unigrams,bigrams,trigrams .

calculate_prob_of_sentence:

מימשנו שתי פונקציות אחת לחישוב לינארי ואחת לחישוב לפלסי ובפונקציה זו קראנו בהתאם לכל אחת מהן לפי סוג הסמוזינג שפונקציה זו מקבלת.

calculate_prob_of_sentence_laplace:

בפונקציה זו חישבנו את הסתברות המשפט לפי הנוסחא הלפלסיאינית:

P=(Triagram_count +1)/(bigram_count+v)

calculate_prob_of_sentence_linear:

בפונקציה זו חישבנו את הסתברות המשפט לפי הנוסחא הזו:

$$P(wn\,|wn-2wn-1\,) = \lambda 1P(wn\,|wn-2wn-1\,) + \lambda 2P(wn\,|wn-1\,) + \lambda 3 \cdot PLaplace(wn\,))$$

לאחר יותר מניסיון קיבלנו תוצאות הכי טובות עבור ערכי למדא אלו :

$0.7=\lambda 1$

$0.2=\lambda 2$

$0.1=\lambda 3$

טריגרם מספקת תלות ספיציפית יותר מכיוון שהיא מחשיבה את שתי המילים הקודמות, מה שהופך את החיזוי שלה למדויק יותר בהקשרים שבהם יש מספיק דאטה. ביגרם, עם רק מילה אחת קודמת, מעניק פחות ספציפיות מטריגרם אבל יותר מאוניגרם, שאינו מתייחס לשום תלות(אינו לוקח בחשבון מה הופיע לפני).

המודל מעדיף להתייחס להקשר הספיציפי כשיש מספיק דאטה,אך במקביל הוא משתמש גם בהקשר הרחב ובתדרי מילים כללים כשדאטה ספיציפי חסר על מנת לבצע חיזוי בגלל כך נתננו ערכי למדא גבוהים לטריגרם אבל גם נתננו ערכי למדא בהתאם לביגראם וגם כן ליוניגרם .

generate_next_token:

מקבלת רצף של טוקנים ,עוברים על כל הטוקנים האפשריים בקורפוס לפי המודל ,ניקח את שני הטוקנים האחרונים ברצף ומוסיפים את הטוקן האפשרי ואז מחשבים את ההסתברות הטריאגראם לפי לינאר ושומרים את ההסתברות המקסימלית ואת הטוקן שמקיים את הסתברות זו.

בסוף המעבר שלנו על כל הטוקנים האפשריים תחזיר הפונקצייה את הטוקן בעל ההסתברות המקסימלית.

במקרה ולמשפט הקלט יש פחות מ 3 טוקנים הוספנו שני טוקני דמי <s>

בתחילת המשפט וכך פתרנו את הבעיה כי במקרה הכי רע אנחנו נקבל משפט יש בו טוקן אחד ואז עם ההוספה זו יהפכו ל 3 טוקנים וכך נוכל לחשב הסתברויות ולטפל במשפט באופן רגיל במודל של הטריאגרם בלי שתהיה לנו בעיה כלשהי.

שלב 2 - קולוקציות

בשלב זה מימשנו את פונקציית **get_k_n_collocations**

פונקצייה זו מקבלת :

k- מספר הקולוקציות שנדרש להחזיר שהם הכי נפוצות בקורפוס מסוים

n- אורך הקולוקציות שנדרש

protocol_type-סוג הפרוטוקול

הפונקצייה מחזירה את k הקולוקציות הכי נפוצות בקורפוס באורך n

על פי מדד הPMI.(בעלות המדד המקסימלי)

לפי הנוסחא הזו:

$$\mathrm{PMI}(w_1, w_2, \ldots, w_n) = \log\left(\frac{P(w_1, w_2, \ldots, w_n)}{\prod_{i=1}^{n} P(w_i)}\right)$$

לאחר מכן השתמשנו ב פקודה()file.write כדי שהפלט יודפס לקובץ טקסט כפי שנדרש ממנו.

שלב 3 - יישום מודלי השפה

1.השלמת הטוקנים התבצעה בעזרת קריאה לפונקציית generate_next_token שמימשנו בשלב 1.

2.חישבנו את ההסתברות לכל אחד מהמשפטים (לאחר השלמתם) בעזרת כל אחד משני מודלי השפה שבנינו בשלב 1.

השתמשנו בפונקציית:calculate_prob_of_sentence(linear)

3.את זה עשינו ע"י השוואה בין ההסתברויות שקיבלנו בסעיף הקודם. הקורפוס שיש לו הסתברות יותר גבוהה הוא זה שיותר סביר שיופיע בו המשפט.

שלב 4 - שאלות סיכום

1.אכן ישנו הבדל בין שני המודלים.

לרוב קיבלנו תוצאות שונות ,זה נובע לפי דעתנו משוני התוכן של כל אחד משני הפרוטוקולים.

.2

כשה-n גודל אנו מצליחים לזהות יותר מה נושא התוכן של כל קורפוס

למשל : שלבנון תנוקה מהרקטות ומה"חיזבאללה - דרכה נוכל להבין שמדובר על המלחמה בין לבנון לישראל.

לא הופתענו, המדדPMI בודק כמה פעמים טוקנים הופיעו יחד ביחס לכמות הפעמים שהופיעו באופן

כללי.

התוצאות שקיבלנו בפעם היחידה שהן הופיעו זה היה יחד לכן קיבלו ציון פ.מ.אי גבוה.

לכן לפי משמעות ה פ.מ.אי זה תואם את הציפיות שלנו.

3.לפעמים קיבלנו משפטים הגיוניים לפעמים לא,הסיבה לכך היא שהרבה פעמים הטריגראם והפיגראם שאנו מחפשים לא יופיעו בקורפוס ויקבלו הסתברות אפס.

בכל זאת,הקורפוס שהמודלים שלנו מתאמנים עליו הינו מאוד קטן ולכן במקרים אלו התשובה תהיה הטוקן הכי נפוץ בקורפוס ,זוהי הסיבה שהרבה פעמים קיבלנו שהמילה החסרה הינה "," או "." שהם טוקנים נפוצים בכל טקסט בדרך כלל.

4.היינו מקבלים משפטים יותר גרועים,מכיוון שב Bigram

תלויים במילה אחת לפני למרות שבtrigram תלויים בשתי מילים לפני ,ובכך הם מספקים הקשר

רחב יותר.

המילה הנוספת בין שני סוגי מודלים אלו מאפשרת למודל להבין טוב יותר את מבנה ומשמעות המשפט,דבר זה מוביל לחזויים טובים ומדוייקים יותר.

trigram יכולים לחזות מילים בצורה יעילה יותר מכיוון שהוא תלוי במילה אחת יותר (שתיים)

מה שהופך אותן לדייקות יותר בהשלמת המילה החסרה במשפטים.

כש n גדול יותר הפרפלקסיטי יקטן מכיוון שהמודל שוקל יותר תלות מה שיכול להוביל

לתחזיות מדויקות יותר.כתוצאה מכך הפרפלקסיטי של המודל נוטה להיות קטנה יותר מכיוון שהוא יותר טוב בלחזות את המילה הבאה ברצף .

Minimizing perplexity is the same as maximizing probability