דו"ח תרגיל בית 2

מודלי שפה – n-grams

שמות המגישים + ת"ז:

עוביידה חטיב, 201278066

מאיה עטואן, 314813494

שלב 1

המחלקה Trigram_LM נוצרה. הבנאי שלה מקבל כקלט רשימה (list) של משפטים, מוסיף לכל משפט שני טוקנים התחלתיים 's_0' ו- 's_1' ושומר אותם כמפוצלים (לטוקנים) בתוך שדה שנקרא **sentences**. שדות אחרים שהמחלקה מכילה הם:

- **tokens_count**: מספר הטוקנים שיש בקורפוס.
- **unique_tokens_count**: מספר הטוקנים השונים שיש בקורפוס.
- **trigrams**: מלון המכיל את כל ה- trigrams שבקורפוס, ולכל אחד את כמות הפעמים שהוא הופיע.
- **bigrams**: מלון המכיל את ה- bigrams שבקורפוס, ועבור כל אחד את כמות הופעותיו.
- **unigrams**: מלון המכיל את ה- unigrams שבקורפוס, ועבור כל אחד את כמות הופעותיו. הרבה פעמים נשתמש במפתחות שלו גם בכדי לעבור על הטוקנים השונים שנמצאים בקורפוס.

את כל השדות המחלקה מאתחלת בעת שנוצרה, כאשר אתחול השלושה האחרונים מתבצע מתוך המשתנה sentences. הדבר העשה ע"י המתודות _trigrams_create, _bigrams_create, _unigrams_create בהתאמה, שעובדות באותה דרך: עבור כל משפט, המתודות מחלצות את כל תתי המשפטים מאורך הטוקנים הרלוונטי שבתוכו. פונקציית ה- _unigrams_create בנוסף לכך סופרת את מספר הטוקנים חוץ מ- 's_0', 's_1' ושמורת את התואצה ב- tokens_count, ואת מספר הטוקנים השונים ששומרת אותם ב- unique_tokens_count.

בשל כך שהמחלקה מקבלת רשימה של משפטים כקלט, המשפטים מחולצים מהקורפוס ומפוצלים ב- ()main למשפטים של מליאות ומשפטים של ועדות, לפני אתחול שני מופעי המחלקה.

שלב 1.1:

הפונקציה 'calculate_prob_of_sentence' מומשה. היא מקבלת כקלט משפט, ומחזירה את לוג ההסתברות שלו. הפונקציה מחשבת את לוג ההסתברות בעזרת נוסחת ה- trigram מההרצאה, כאשר ההסתברויות מחושבות באמצעות MLE עם החלקת לפלס כנדרש. הפונקציה מוסיפה למשפט שני טוקני דמה בתחילתו, והחישוב של ההסתברות מתבצע ע"י כך שהיא עוברת על כל טוקן במשפט החל מהשלישי, שולחת אותו ביחד עם שני הטוקנים שלפניו לפונקציית עזר בשם 'calculate_prob_of_token', שבתורה מחזירה את ההסתברות של הטוקן. הפונקציה סוכמת את הערכים המוחזרים מפונקציית העזר עבור כל הטוקנים ומחזירה את התוצאה.

פונקצית העזר 'calculate_prob_of_token' מקבלת כקלט משפט בעל 3 טוקנים, ומחשבת את ההסתברות של הטוקן השלישי כתלות בשני הראשונים, תוך שהיא משתמשת בנוסחת ה- trigram.

מתמטית, אופן פעולת שתי הפונקציות הוא : בהינתן המשפט $s = w_0 w_1 w_2, .., w_{n-1}$, לתחילת המשפט מתווספים שני טוקני דמה s_0, s_1, כך שהמשפט הופך להיות $s = s_0 s_1 w_0 w_1 w_2, .., w_{n-1}$, וההסתברות שלו מחושבת באמצעות הנוסחה הבאה:

$$P(s) = \sum_{w_i \in s} \log(P(w_i | w_{i-2} w_{i-1}))$$

כאשר:

$$P(w_i | w_{i-2} w_{i-1}) = \lambda_1 * P(w_i | w_{i-2} w_{i-1}) + \lambda_2 * P(w_i | w_{i-1}) + \lambda_3 * P(w_i)$$

כך ש-

$$P(w_i | w_{i-k}, \dots, w_{i-1}) = \frac{Count(w_{i-k}, \dots, w_i) + 1}{Count(w_{i-k}, \dots, w_{i-1}) + (\#\ of\ unique\ tokens)}$$

המקדמים $\lambda_1, \lambda_2, \lambda_3$ נבחרו להיות 0.85, 0.14, 0.01 בהתאמה. המקדמים נבחרו על סמך ניסוי שהשווה בין שלשות שונות של מקדמים באחוז הצלחתם בניבוי טוקן חסר במשפט, בנוסף לשימוש ב- Perplexity כאינדיקטור נוסף. באשר לניסוי, הוא כלל 500 משפטים שנבחרו באופן אקראי מקורפוס הוועדות, ובכל אחד מהם 10% מהטוקנים מוסכו. הניבוי של הטוקנים הממוסכים התבצע ע"י הפונקציה 'generate_next_token' שמסתמכת בניבוי שלה על הסתברות המשפט שמחושב ע"י הקריאה ל- 'calculate_prob_of_sentence '. הצעד הזה של בחירת המקדמים נעשה בסוף (בהתחלה השתמשנו במקדמים שרירותיים),

ובחירת המקדמים שישתתפו בניסוי נעשתה על סמך אבחנות שהתקבלו במהלך כתיבת הקוד, בנוסף ל- "מיני" ניסויים שנעשו במטרה ללמוד את התרומה של המקדמים ואיך הגדלת\הקטנת כל אחד מהם משפיעה על התנהגות פונקציית הניבוי. שני הדברים הנ"ל צמצמו את טווח האופציות של כל מקדם. ניתן לסכם את האבחנות בכך שהמקדם של ה- unigrams צריך להיות קטן יחסית, ושהמקדמים (החל ממקדם ה- trigrams) צריכים להיות גדולים אחד מהשני בהפרש משמעותי. האבחנה הראשונה התקבלה בשל כך ששמנו לב שמקדם גבוה עבור ה- unigrams נתן משקל יתר לטוקנים נפוצים כמו "," ו- "אני", גם כשהמיקום שלהם לא תואם מבחינת תחביר ושפה, וגם לא מבחינת הסתברות של משפט (למשל, האלגוריתם ניבא את הטוקן "," אחרי הטוקן "עמיר" למרות שהופעת הטוקן "פרץ" אחרי "עמיר" יותר שכיחה בקורפוס). לגבי האבחנה השנייה, היה אפשר לשים לב לכך שיכולת הניבוי של ה- bigrams יותר טובה ביחס ל- unigrams ושל ה- trigrams יותר טובה משניהם. המשקל הגבוה של ה- bigrams יחסית ל- unigram ניתן על סמך כך שהרבה צירופי טוקנים באים דווקא כזוגות, דוגמת: "חוות דעת", "אמר :", ושמות משפחות של אנשים מפורסמים שבאים בעקבות השמות הפרטיים שלהם. מקדם גבוה ל- bigrams נתן לאלגוריתם להצליח לנבא את המילה השנייה בצירופים מסוג זה. מצד שני, מקדם גדול מדי ל- bigrams פגע בצירופים הפחות נפוצים, במיוחד בצירופים בהם הטוקן השני פחות מזווג עם הטוקן הראשון מאשר טוקן אחר. למשל, בשל שכיחות הצירוף "במגזר הערבי" בקורפוס, האלגוריתם נכשל בלנבא את המילה "הפרטי" כהמשך למשפט "והן עם העובדים במגזר [*]", מכאן קיבלנו מסקנה שקונטקסט יותר גדול נדרש על מנת להבדיל בין שני המקרים ("במגזר הערבי" ו- "במגזר הפרטי"). בנוסף לכך, ובשל העובדה שמספר המופעים של trigrams קטן משמעותית ממספר ה- bigrams (למשל, "העובדים במגזר הפרטי" לעומת "במגזר הפרטי"), הגענו למסקנה שהמשקל של ה- trigrams חשוב שיהיה גבוה בהרבה מה- bigrams.

שלב 1.2:

הפונקציה 'generate_next_token ' מומשה. היא מקבלת כקלט מחרוזת (רצף של טוקנים), ומחזירה tuple בעל שני אלמנטים: הטוקן בעל ההסתברות הכי גבוהה שיהיה ההמשך של המחרוזת, ולוג ההסתברות שלו. לשם כך, הפונקציה עוברת על כל הטוקנים השונים (או באופן יותר מדויק, על ה- unigrams, מכיוון ש- unigrams וטוקנים שקולים במקרה שלנו), מוסיפה את הטוקן למחרוזת ושולחת את התוצאה של החיבור לפונקציה 'calculate_prob_of_sentence' שבתורה מחזירה את לוג ההסתברות של המשפט.

הפונקציה תחזיר את הטוקן שהתקבל עבורו לוג ההסתברות הגבוה ביותר ביחד עם לוג ההסתברות.

למרות שהקפדנו על מימוש שתי הפונקציות 'generate_next_token' ו- 'calculate_prob_of_sentence' באופן האופטימלי, בנוסף למאמץ שהושקע בבחירת המקדמים, הפונקציה כשלה הרבה פעמים בניבוי הטוקן בצורה נכונה. מקרים בולטים בהם הפונקציה נכשלת בניבוי הם בטוקנים שהם חלק מקולקציות פחות נפוצות, במיוחד כאשר יש טוקנים יותר שכיחים כהמשך של הקולקציה. דוגמה לכך הוא כישלון האלגוריתם בניבוי הטוקן "סעד" כחסר במשפט "בבקשה , חבר הכנסת אחמד [*] . ", בשל כך שהקולקציה "אחמד טיבי" יותר שכיחה. מכיוון שאחמד טיבי גם כן חבר כנסת, גם קונטקסט יותר גדול כמו ה- trigram (במקרה הזה הוא "הכנסת אחמד [*]") לא עזר. סוג מקרים אחר הוא כאשר הטוקנים שלפני הטוקן הנדרש להשלים לא נותנים קונטקסט כלשהו, בחלק מהמקרים הללו, דווקא הטוקנים שבאים אחרי הם אלה שכן היו יכולים לעזור בניבוי הטוקן החסר, דבר שנתקלנו בו כאשר השתמשנו בפונקציה לניבוי טוקן ממוסך. דוגמה לכך היא בניבוי המילה "השר" במשפט "... אני לא [*] לביטחון הפנים ... ", שבו הטוקנים אחרי הטוקן הממוסך הם הרלוונטיים שיכולים לתת הקשר.

בחלק מהמקרים בהם הפונקציה נכשלה בניבוי היה ניתן לשים לב לכך שהניבוי של הפונקציה היה טוקן שכן נתן משמעות למשפט למרות היותו לא הנכון. למשל, בניבוי הטוקן "מנגנון" במקום "עוד" במשפט " כדי להתגבר על הפער צריך לייצר [*] , אבל ..."

מצד שני, הפונקציה כן הצליחה להתמודד עם השלמת הטוקן החסר בקולקציות נפוצות כמו הטוקן "דעת" כהמשך למשפט "לקבל חוות [*]", וגם בהתמודדות עם טוקנים פחות שכיחים כהמשך של קולקציות מסוימות מאורך 2 אם ההקשר הרחב יותר של ה- trigram עזר בכך. דוגמה לכך שהוזכרה קודם היא בהצלחת הפונקציה בניבוי הטוקן "הפרטי" כהמשך למשפט "העובדים במגזר [*]" למרות היות שני הצירופים "במגזר הערבי" ו- "במגזר החרדי" יותר נפוצים בקורפוס בהשוואה ל- "במגזר הפרטי".

הצעות שלדעתנו יכולות לשפר את ביצועי הפונקציה: שימוש ב- n-grams יותר גבוהים כגון Four-grams, Five-grams, שימוש בקורפוס יותר גדול שיכיל יותר קולקציות ובכללי יהווה מדגם גדול יותר, בנוסף להתחשבות בטוקנים שאחרי הטוקן החסר ולא רק באלה שלפניו במקרים שטוקנים כאלה כן נמצאים (למשל, כאשר מתבקשים לחזות את טוקן ממוסך שלא בסוף המשפט).

שלב 2

שלב 2.1:

הפונקציה 'get_k_n_t_collocations' מומשה. הפונקציה מחזירה אוסף של קולקציות העונות על הדרישות הבאות שמתקבלות כקלט: מספרם הוא k, אורך כל אחת מהן הוא n, ומספר המופעים של כל אחת לא פחות מ- t בקורפוס מסוים, שהוא pandas dataframe. כמו כן, הפונקציה מקבלת כקלט את סוג המדד שלפיו מחליטים איזה k מבין כל הקולקציות העונות על הדרישות צריך לבחור.

הפונקציה מתחילה בכך שהיא מחלצת את כל הקולקציות מאורך n הנמצאות בקורפוס. הדבר מתבצע ע"י כך שהיא עוברת על כל המשפטים בקורפוס, ועבור כל משפט שאורכו לפחות n, היא מחלצת את כל הקולקציות באורך n השונות שבתוכה (כל המחרוזות של טוקנים רצופים בגודל n). תוך כדי כך, היא סופרת את כמות הופעות כל קולקציה בפרוטוקולים השונים, ואת הקולקציות המופיעות בכל פרוטוקול וכמות הופעתם. הספירה מאוחסנת בעזרת שני מלונים, אחד לפי הקולקציה והשני לפי הפרוטוקול. הפונקציה, אח"כ, מסננת את הקולקציות ומשאירה רק את אלה שמספר הופעתם בכלל הפרוטוקולים הוא לפחות t. אחרי הסינון, הפונקציה מסדרת את הקולקציות מהכי נפוצה לפחות לפי המדד המועבר לפונקציה, שני המדדים הם "Frequency" ו- "TF-IDF", כאשר הראשון מסדר אותם לפי כמות ההופעות הכללי, והשני לפי הנוסחה הנתונה, שמתחשבת במספר הקולקציות השונות בפרוטוקול ואת מספר הפרוטוקולים השונים שבהם הקולקציה הופיעה. אם לפונקציה מועבר מדד אחר מהשניים הנ"ל היא מצגיה Value Error ומדפיסה הודעה בהתאם. אחרי שלב הסידור, הפונקציה מחזירה את ה- k קולקציות הראשונות בסדר. במקרה שמספר הקולקציות הוא פחות מ- k, היא מחזירה את כולם.

שלבים 2.2 – 2.4:

הקוד שמדפיס את הדרוש נמצא ב- main ומשתמש בפונקציה מהשלב 2.1 לשם כך. הפלט מצורף כקובץ txt כנדרש.

שלב 3

שלב 3.1:

הפונקציה 'mask_tokens_in_sentences' מומשה. היא מקבלת כקלט רשימה (list) של משפטים ומספר x בטווח [0,100]. הפונקציה ממסכת x% מהטוקנים במשפט (מחליפה אותם ב- [*]). ומחזירה משפט חדש עם הטוקנים הממוסכים. במקרה ש- *x% * (אורך*

המשפט) לא מספר שלם היא מקרבת אותו לשלם הכי קרוב, ובשל כך, משפט בעל פחות מ-5 טוקנים (במקרה של המשפטים שלנו, משפט בעל 4 טוקנים) אף טוקן בו לא ימסך. כמו כן, שמנו לב לכך שהפונקציה ()round מעגלת את המספר 0.5 ל- 0, וכתוצאה מכך גם משפט בעל 5 טוקנים לא ימסך, לכן ועל מנת לעמוד בדרישת התרגיל של למסך לפחות טוקן אחד בכל משפט, הוספנו מקרה מיוחד בו אם התוצאה של *x% * (אורך המשפט)* היא 0.5 היא תעוגל ל- 1. במקרה שהמספר x לא בטווח [0,100] הפונקציה מציגה Value Error ומדפיסה הודעה בהתאם. את הטוקנים שהיא ממסכת הפונקציה בוחרת באקראי.

שלב 3.2:

הקוד מומש ב- main, והפלט נמצא בשני קבצי txt מצורפים כנדרש.

הקוד מתחיל בחילוץ המשפטים בעלי 5 טוקנים לפחות מתוך קורפוס הועדות כנדרש בתרגיל. הוא מזהה אותם ע"י כך שיש בתוכם 4 רווחים לפחות. הקוד אח"כ בוחר 10 משפטים באקראי, מדפיס אותם לקובץ 'original_sampled_sents.txt'. לאחר מכן, המשפטים נשלחים לפונקציה 'mask_tokens_in_sentences ', אשר מחזירה את המשפטים כך ש- 10% מהטוקנים בכל אחד ממוסכים, והמשפטים הממוסכים מודפסים לקובץ 'masked_sampled_sents.txt'. במידה ומספר המשפטים בעלי לפחות 5 טוקנים הוא פחות מ- 10, הקוד מדפיס Value Error עם הודעת שגיאה.

שלב 3.3:

הקוד מומש ב- main, והפלט נמצא בקובץ txt המצורף כנדרש.

עבור כל משפט, הקוד מדפיס את המשפט המקורי והמשפט אחרי המיסוך. לאחר מכן, הוא מאתר את כל סימני "[*]" אשר מופיעים במשפט, עבור כל אחד החל מהראשון (הימני ביותר) הוא שולח לפונקציה 'generate_next_token ' של מודל הועדות את תת-המשפט עד אותו [*], והפונקציה בתורה מחזירה טוקן, הקוד מחליף את [*] בטוקן המוחזר. הקוד לאחר מכן מדפיס את המשפט החדש שהתקבל בעקבות ה- "אי-מיסוך", את הטוקנים שהחליפו את ה-"[*]"-ים, ומחשב ומדפיס את ההסתברות עבור המשפט בכל אחד משני המודלים באמצעות הפונקציה 'calculate_prob_of_sentence ' של כל אחד מהם.

שלב 3.4:

הקוד מומש ב- main.

הנוסחה שהשתמשנו בה לחישוב ה- Perplexity היא אותה נוסחה שהוזכרה בהרצאה עם השינוי שכעת החישוב כלל רק את הטוקנים הממוסכים. מתמטית, נסמן ב- s את המשפט, וב- $w_1, \dots, w_n$ את הטוקנים הממוסכים. כמו כן, נסמן ב- s[j:i] את החלק במשפט שבין הטוקנים j ו- i-1. אז הנוסחה היא:

$$\log(Perplexity) = log\left(\prod_{i=1}^{n} \frac{1}{P(w_i|s[:i]}\right)^{\frac{1}{n}}$$

$$= -\frac{1}{n}\sum_{i=1}^{n} \log(P(w_i|s[i-2:i])$$

$$\rightarrow Perplexity = 2^{\log(Perplexity)} = \mathbf{2}^{-\frac{\mathbf{1}}{\boldsymbol{n}}\sum_{\boldsymbol{i}=\mathbf{1}}^{\boldsymbol{n}} \mathbf{log}(\boldsymbol{P}(\boldsymbol{w_i}|\boldsymbol{s}[\boldsymbol{i}-\mathbf{2}:\boldsymbol{i}])}$$

החישוב הנ"ל התבצע עבור כל אחד מ- 10 המשפטים, והתמצע ע"י כך שהתחלק ב- 10 על מנת לקבל את ה- perplexity הממוצע. התוצאה שקיבלנו הייתה **673.35**.

שלב 4

שאלה 4.1

הקולקציות הנפוצות לפי מדד התדירות ברובן קולקציות כלליות בשפה אשר יכולות להופיע בכל קורפוס ללא קשר לתוכן שלו. דוגמאות לכך: "את זה", "אני רוצה". עם זאת ככל שאורך הקולקציה יותר גדול אפשר לשים לב ליותר קולקציות שמעידות על התוכן של הקורפוס במיוחד בקולקציות מאורך 4, כמו בקולקציה "חברי חברי הכנסת ,". כמו כן, ניתן לשים לב שהקולקציות שנלקחו מקורפוס המליאות יותר רלוונטיות לתוכן מאשר אלה שנלקחו מקובץ הוועידות, כאלה המכילות התייחסות לחברי הכנסת ולתפקידים רשמיים. למרות שאפשר למצוא קולקציות שמעידות על התוכן הכללי שבו הקורפוס עוסק, קשה לדעת מהם על דברים יותר ספציפיים כגון הנושאים המדוברים בתוכו.

התוצאות תאמו במידה רבה לציפיות שלנו של לקבל קולקציות שכיחות בשפה, במידה פחות לציפיות של לקבל מספר לא מעט של קולקציות שאמורות להיות שכיחות בדיוני הכנסת כמו "ראש הממשלה", "חבר הכנסת", ו- "פתיחת הדיון". דבר אחר שנראה לנו הגיוני הוא הנטייה של קולקציות המתייחסות לתפקידים רשמיים להופיע יותר בפרוטוקולי המליאות מאשר

בפרוטוקולי הוועידות, לאור היות השיח במליאות יותר פורמלי ופחות ספונטני מזה שבוועידות. מה שלא הופיע בתוצאות למרות היותו צפוי הוא קולקציות הנוגעות לשמות פרטיים של חברי כנסת, במיוחד אלה מתוכם שכיהנו תקופה רבה במהלך השנים מהם נלקחו הפרוטוקולים, שמות של מפלגות גדולות, שמות ערים ואזורים שונים בישראל, כמו גם קולקציות הנוגעות לנושאים שתמיד נמצאים על הפרק כמו ההתנחלויות והאיום הביטחוני.

שאלה 4.2

הקולקציות הנפוצות ע"פ מדד TF-IDF יותר נוגעות לתוכן הקורפוס מאשר הקולקציות הנפוצות ע"פ מדד התדירות ופחות כלליות. הן נוטות יותר להכיל אזכורים לכנסת ולתפקידים פוליטיים. כמו כן, בניגוד לקולקציות המתקבלות ע"י מדד התדירות, הקולקציות של מדד ה- TD-IDF מכילות יותר התייחסות לנושאים ספציפיים כמו הקולקציות "בית החולים" ו- "נמל התעופה". דבר שניתן לשים לב אליו הוא שיש חלק מהקולקציות שמתייחסות לשנים דוגמת "התשס"ו – 2006" ו- "התש"ס – 2000", אשר יכולות לתת גם מידע לגבי הזמן בו התקיימו הדיונים, או לזמן התרחשות האירועים בהם דנים.

התוצאות כן תאמו לציפיות שלנו של לקבל קולקציות שנוגעות לנשואים בהם דנים בכנסת יותר מאשר בקולקציות לפי מדד התדירות בשל הנטייה של מדד TD-IDF להבליט קולקציות ייחודיות לפרוטוקול ספציפי. אותה צפייה ואותו הסבר נוגע גם להופעת קולקציות המתייחסות לשנים, מכיוון שסביר להניח ששנים אמורות להופיע רק בפרוטוקולים של אותה שנה או בפרוטוקולים מטווח שנים קטן סביב אותה שנה, בנוסף להיותם שכיחים בפרוטוקולים האלה. שני הדברים אמורים לתרום לערך ה- TF-IDF של הקולקציה. בדומה לקולקציות הנפוצות על פי מדד התדירות, גם כאן לא התקבלו קולקציות המכילות שמות של אנשים, מפלגות, ערים ומקומות גאוגרפיים, דבר שהיה מנוגד לצפיות שלנו.

שאלה 4.3

כן, יש הבדלים בין שני המדדים:

1. הקולקציות הנפוצות לפי מדד TF-IDF יותר קשורות לתוכן הקורפוס מאשר הקולקציות הנפוצות לפי מדד התדירות שיותר מכילות קולקציות כלליות היכולות להופיע בתדירות גבוהה בכל קורפוס ללא קשר לתוכן שלו.
2. הקולקציות הנפוצות לפי מדד TF-IDF יותר ספציפיים ויכולים לספק מידע לגבי נושאים בהם הפרוטוקולים מתעסקים לעומת הקולקציות של מדד התדירות, שגם

אם מייחסות לתוכן הקורפוס, הן נוטות להיות יותר כלליות ("בית החולים" ו- "נמל התעופה" לעומת "חברי הכנסת").

3. בקולקציות של מדד TF-IDF נוטים להופיע מספרים (רובם מתייחסים לשנים אך לא רק) יותר מאשר בקולקציות של מדד התדירות.
4. קולקציות של מדד TF-IDF נותנות מידע לגבי התוכן של הקורפוס גם כאשר הן מאורך קצר, בניגוד לקולקציות מדד התדירות שהמידע על התוכן יכול להופיע רק בארוכות מתוכן.

שאלה 4.4

הגדלת או הקטנת הסף t, שקובע את מספר המופעים המינימלי של הקולקציה בקורפוס, **לא אמור לשנות את הקולקציות המוחזרות ע"י מדד התדירות**, כל עוד הוא לא גורם לסינון שמשאיר מספר משפטים שהוא פחות מ- n, מפני שהקולקציות בעלות התדירות הגבוהות ימשיכו לעבור את הסף, כמו גם שהסדר שלהן לא ישתנה. השינוי היחיד שיכול להיות לקולקציות המודפסות כתוצאה משינוי הערך של t, הוא כאשר מגדילים את t עד לסף שיש פחות מ- n משפטים שעונים עליו, ובכך השינוי יהיה ע"י כך שרק חלק מהמשפטים מבין אלה שהודפסו קודם יודפסו כעת. או להפך, אם הערך t הקודם לא אפשר ל- n משפטים לעבור אותו והקטנתו מאפשרת ליותר משפטים לעבור את הסינון ובכך להיות מודפסות בנוסף לאלה שהודפסו קודם. השינויים האלה תקפים לכל האורכים השונים של הקולקציות (2,3,4), אולם חשוב לציין, שהסבירות של העלאת הערך של t לפגוע בקולקציות מאורך גדול (כמו אורך 4) יותר גדולה מאשר לפגוע בקולקציות מאורך קטן יותר בשל היותם פחות נפוצים וגם בגלל היות כל קולקציה מאורך 4 מכילה 2 קולקציות מאורך 3 ו- 3 קולקציות מאורך 2, וכך גם לגבי קולקציות מאורך 3 המכילות כל אחת 2 קולקציות מאורך 2. במלים אחרות, לא יכול להיות מקרה בו יש כמות של קולקציות מאורך 4 שעונות על הסף יותר מאשר מספר הקולקציות מאורך 3 או 2 שעונות עליו.

לעומת זאת, **הקולקציות השכיחות המתקבלות על פי מדד TF-IDF עשויות כן להיות מושפעות מהגדלה או הקטנה של הערך של t**. למרות שהמדד, בדומה למדד התדירות, מתחשב בשכיחות הקולקציה על פני הקורפוס, אך הוא לא הקריטריון היחיד שנלקח בחשבון, והרבה קולקציות מקבלות הועדפה על ידיו בשל היותן שכיחות במספר מועט של קורפוסים או בשל היות הקורפוסים שבהם הן שכיחות בעל מספר מועט של קולקציות. קולקציות כאלה לא בהכרח מהכי שכיחות (לפי מספר הופעתם הכולל), לכן העלאת הסף יכול לגרום להן להיות מודחות, והקטנתו יכול לאפשר לקולקציות דומות בעלות ערך TF-IDF גבוה מהקולקציות הקיימות להיות מוכלות ולקבל העדפה על פניהם. האפקט הזה שתואר צפוי

להיות בעל השפעה יותר משמעותית עבור צירופים מאורכים גדולים לעומת צירופים מאורכים קטנים בשל היות הקולקציות הארוכות פחות נפוצות, ולכן ההפרשים בין כמויות ההופעות של הקולקציות בפרוטוקול ($f_{t,d}$) יותר קטן ופחות משמעותי בחישוב ה- TF-IDF, מה שגורם לערכים של $\sum_{t' \in D} f_{t',d}$ ו- $|\{d \in D: t \in d\}|$ לשחק תפקיד גדול יותר בקביעת ערך ה- TF-IDF. מכאן, אנחנו מצפים **שההשפעה והשינוי בקולקציות המודפסות יהיה גדול יותר בקולקציות מגודל *n=4* לעומת *n=3*, וגדול יותר בקולקציות מגודל *n=3* לעומת הקולקציות מגודל *n=2***.

שאלה 4.5

חלק מהמשפטים שהתקבלו בשלב 3.3 הם הגיוניים. בחלק של המשפטים הלא הגיוניים יש מקרים בהם הטוקן המשלים היא כן נכון כמשמעות אך לא כתחביר. דוגמה לכך במשפט הבא:

" הראשונה היא אזרחי **מדינת** שתפסה את השטח ..."

תופעה אחרת שאפשר לצפות בה במשפט הנ"ל, ובאופן יותר קיצוני במשפטים אחרים היא שהחלק של המשפט עד המילה הוא הגיוני אך לא לגבי המשפט כולו. כמו כן, ניתן לשים לב שממשפט ממסוך אשר הטוקנים הממוסכים בו קרובים אחד לשני נוצר משפט שלרוב לא הגיוני.

שאלה 4.6

יש משפטים שהאלגוריתם מצליח לחזות את הטוקן החסר בהם, כפי שהוזכר קודם בפירוט בשלבי 1.1 ו- 1.2. במשפטים שהאלגוריתם לא מצליח לחזות את הטוקן החסר בהם, לרוב גם אם הוא יוצר משפט הגיוני, הטוקן המשלים לא קרוב לאמיתי. דוגמה לכך היא במשפט הבא, שבו הטוקן האחרון מוסך, וההשלמה למרות שיצרה משפט בעל משמעות שקרובה לשל המקורי, היא בעצמה לא קרובה לטוקן האמיתי.

המשפט המקורי: "אבל לוותר על 100 מיליון **?**"

המשפט אחרי המיסוך: "אבל לוותר על 100 מיליון **[*]**"

המשפט החדש: "אבל לוותר על 100 מיליון **שקל**"

למרות זאת, יש מקרים שכן המשמעות של הטוקן קרובה מאד לאמיתי, כמו המקרה הבא:

המשפט המקורי: "בשלב **הזה** אני רוצה שתחשבו על רעיונות ."

המשפט אחרי המיסוך: " בשלב **[*]** אני רוצה שתחשבו על רעיונות ."

המשפט החדש: "בשלב **זה** אני רוצה שתחשבו על רעיונות ."

שאלה 4.7

נוסחת ה- Perplexity משמשת להערכת ביצוע המודל על הטקסט בלחזות את המילה הבאה במשפט. התוצאה המספרית שמתקבלת מהנוסחה מצביעה על רמת "ההפתעה" של המודל כאשר הוא נתקל במילה חדשה. התוצאה שקיבלנו היא **673.35**, שהוא ערך גבוה יחסית, המצביע על כך שהמודל לא כל כך טוב בחיזוי הטוקנים הבאים במשפט ומתקשה להבין את ההקשר של המשפט, אולם הוא לא אקראי לגמרי ויכול להצליח בניבוי משפטים טריוויאליים, דבר שאכן משתקף בתוצאת שקיבלנו בשלב 3.3.

שלב 4.8

התוצאה שהתקבלה, 673.35, מעידה על ביצוע לא טוב יחסית, כפי שהוסבר בסעיף הקודם. הסיבות לכך יכולות להיות:

- גודל המדגם, שעל אף שכללנו עשרות פרוטוקולים, הוא עדיין נחשב לקטן ולא מכסה מספר רב של קולקציות, או לא מכיל מספיק מופעים של קולקציות מסוימות.
- המודל של ה- *trigram* לא מספיק על מנת ללמוד את ההקשר של המשפט, ונצטרך להשתמש ב- *n-grams* גבוהים יותר, כמו *four-grams* ו- *five-grams*.