יבש 1 - NLP

מגישים: אמיר בלדר - 204179659, רועי גנץ - 204506349

חלק 1

1. בעיית ה- OOV הינה בעיה מוכרת, כיצד נוכל לסווג מילים שלא ראינו קודם לכן באימון?
ישנם מספר סוגים של "משפחות" מאפיינים שיכולים לעזור בכדי לנתח בעיית OOV.

אנחנו יכולים להסתכל על משפחות מאפיינים שבהן יש משמעות לחלקים מסוימים במילה, לדוגמא:

א. שימוש באותיות גדולות/קטנות - מה שיכול לרמז על שם של דבר כלשהו.
ב. שימוש בתחיליות/סיומות שחוזרות על עצמן.
ג. שימוש במספרים בתוך מילים.

שימוש במאפיינים מעין אלה יהווה פתח לניתוחן של מילים רבות שלא ראינו עד כה, ולהוות לפחות ניתוח ראשוני ובסיסי עבור אלה בעיקר השימוש בתחיליות/סיומות שחוזרות על עצמן יכול להיות יעיל, שכן לרוב מדובר בתוספת למילה אחרת, שיתכן שכן הופיעה מוקדם יותר במודל.

2.
א. מדובר בעצם במרחב האפשרויות של וקטורי הייצוג. מרחב זה יהיה מורכב מכמות התיוגים והמילים האפשרויות. לכל מילה יש 100,000 אפשרויות, ולכל תיוג 25 אפשרויות ולכן:

$$|(word_{t-2}, tag_{t-2}, word_{t-1}, tag_{t-1}, word_t, tag_t)|$$
$$= |100{,}000 * 25 * 100{,}000 * 25 * 100{,}000 * 25| = 100{,}000^3 * 25^3$$
$$= 1.5625 * 10^{19}$$

כלומר, שמספר המאפיינים האפשריים הוא $1.5625 * 10^{19}$.

ב. במודל HMM רגיל, הפרמטר $p(x_i|y_i)$ מחושב באופן הבא - $\frac{count(x_i,y_i)}{count(y_i)}$. אולם, במודל החדש, x_i מוחלף בווקטור מאפיינים ארוך. עקרון הפעולה של חישוב ההסתברות המותנית נותר בעינו אך כעת פועל על וקטור ממימד גבוה מאוד.

כעת כל x הינו שלוש מילים ושלושה תיוגים, $< x_i, x_{i-1}, x_{i-2}, t_i, t_{i-1}, t_{i-2} >$, במקום מילה יחידה ותיוג. כלל האפשרויות לשישייה שכזו גדולה משמעותית מהכמות אשר מופיעה בקורפוס. לכן, הספירה עבור הרבה קומבינציות תתאפסנה על אף שהן סבירות בשפה. אמנם במודל שפה יש הרבה קומבינציות לא אפשריות, אך בשל הגודל של הקורפוס, גם קומבינציות שכן אפשריות, יקבלו בחלקן הסתברות אפס - דבר לא רצוי. כמו כן, חישוב שכזה עשוי להיות פחות יעיל מאשר הפעולה המקבילה על מודל עם ייצוג פשוט (תלוי באופן הגדרת הספירה).

באופן כללי, באלגוריתמים של למידה, כדי לאפשר למידה אפקטיבית ויכולת הכללה טובה, כלל אצבע שימושי הוא שמספר הפרמטרים יהיה קטן משמעותית ביחס למספר הדוגמאות. זה לחלוטין לא המקרה עבור מודל זה ולכן הערכת הפרמטרים לא תהא אפקטיבית.

3. נתחיל מלציין עובדה חשובה - פונקציית ה-exp הינה פונקציה מונטונית עולה, שהינה גדולה מאפס. על כן, השימוש בה, יחד עם הנרמול במכנה, משאיר את התוצאה בתחום [0,1], כיאה להסתברות. בנוסף, בשל היותה מונוטונית עולה, השימוש בה לא ישנה את ה"מגמה" ואת מציאת הכיוון הנכון של הגרדיאנט לו נזדקק לעדכון המשקלים בהמשך. יתר על כן, האקספוננט הינה פונקציה גזירה, תכונה הכרחית לצורך שימוש בשיטות אופטימיזציה מבוססת גרדיאנט.

4. בהינתן בחירה בין שימוש בייצוג בינארי עבור מאפיינים לבין ייצוג בעזרת מספרים, נעדיף להשתמש בייצוג בינארי, שכן הוא עשיר יותר. בייצוג מספרי, הפרמטר הנלמד הוא מספר בודד (ממימד אחד). הייצוג הבינארי לעומתו מגוון יותר - הפרמטר הנלמד הוא וקטור משקולות ארוך. לכן, מודל המבוסס על ייצוג וקטורי יהיה עשיר יותר ויאפשר להתחשב ביחסים בין המאפיינים השונים. לעומת זאת, בייצוג מספרי, היות והפרמטר הינו מספר, הגדלת w תגדיל את ההסתברות לכל קומבינציות התכוניות, ולהיפך. נציין שעבודה עם ייצוג מספרי יחיד, יאלץ הגדרת סדר על התכוניות, דבר לא אפשרי. אולם, יש לציין כי מודל ייצוג מספרי עדיף משיקולי מימדיות הייצוג, שכן הוא תמציתי בהרבה (אך משלמים על כך ב-capacity של המודל, כפי שהסברנו לעיל).

5. נבחן את הנוסחא:

$$p(x_i = x|y_i = y) = \frac{p(x_i = x, y_i = y)}{p(y_i = y)} = \frac{p(x_i = x, y_i = y)}{\sum_{\hat{x}\in\tilde{X}} p(x_i = \hat{x}, y_i = y)} = \frac{exp(w \cdot f(x, y))}{\sum_{\hat{x}\in\tilde{X}} exp(w \cdot f(\hat{x}, y))}$$

נשים לב כי החישוב של המכנה הוא בעל אפשרויות רבות מאוד:

$$\sum_{\hat{x}\in\tilde{X}} exp(w \cdot f(\hat{x}, y))$$

כלומר, החישוב של המכנה ידרוש מעבר על המון אפשרויות שונות, ביצוע exp על כל אחת מהן ולאחר מכן סכימתן. כפי שציינו קודם לכן, מדובר על סדרי גודל גדולים למדי של אפשרויות שונות, ועל כן, החישוב עצמו יארך זמן רב מאוד. לכן, ככל הנראה שלא יהיה משתלם להשתמש בנוסחא המדוברת.

6. עתה אנו מסתכלים על נוסחאות אחרות:

$$p(x, y) = \prod_{i=1:n} p(y_{i+1}|y_i, x) =\sim \prod_{i=1:n} p(y_{i+1}|y_i, x_i) = p(y|x) \tag{3}$$

$$p(y_{i+1} = y|x_i = x, y_i = y') = \frac{exp(w \cdot f(x, y', y))}{\sum_{\hat{y}\in Y} exp(w \cdot f(x, y', \hat{y}))} \tag{4}$$

$$\sum_{\hat{y}\in Y} exp(w \cdot f(x, y', \hat{y}))$$

כלומר שבמקרה הזה אנו מסתכלים על המכנה:

כאשר Y הוא קטן משמעותית מאשר X. Y מהווה את מרחב התיוגים השונים, כאשר הדוגמא שראינו בשאלה 2, גודל המרחב היה 25 תיוגים, ובתרגיל הבית שבו אנו השתמשנו במרחב תיוגים סטדנדרטי הוא היה בגדול 45. על כן, מדובר עתה בבעיה שניתן באמת לחשבה בצורה סבירה.

7. ראשית, נציין כי מודל ה- HMM הינו מודל גנרטיבי ומודל ה- MEMM הינו דיסקרימינטיבי. ככלל אצבע, מודל גנרטיבי משערך את המודל ההסתברותי של כל אחד מה-class-ים האפשריים בבעיה, בעוד שמודל דיסקרימנטיבי משערך את גבולות ההכרעה בין ה-class-ים האפשריים (decision boundaries). אם כן, שימוש במודל גנרטיבי יכול להיות טוב יותר למשימות מסוימות, לדוגמא אם נרצה ליצור משפטים. בנוסף, HMM הוא מודל פשוט הרבה יותר להבנה, הן בשל הסיבה שהוא משערך את ההסתברות של כל class ישירות והן בשל אופי החישוב הפשוט. באלגוריתמי למידת מכונה, ההבנה של תוצאות המודל הינה מאוד חשובה, וכאמור במודל ה-HMM קל יותר לנתח את התוצאות, להבין כיצד המודל פועל ועל סמך מה מבסס את תוצאותיו.

חלק 2

1. נציב את משוואה 7 לתוך משוואה 8 ונפתח את הביטוי:

$$\theta^* = argmax_\theta \sum_{(x,y)\in D} \log\big(p(y|x)\big) = argmax_\theta \sum_{(x,y)\in D} \log\left(\frac{e^{v_x * v_y}}{\sum_{z\in v\setminus\{x\}} e^{v_x * v_z}}\right) =$$

$$argmax_\theta \sum_{(x,y)\in D} \mathrm{v_x} * v_y - log \sum_{z\in v\setminus\{x\}} e^{v_x * v_z}$$

המעבר הראשון הינו הצבה של משוואה 7 ואילו המעבר השני הינו שימוש בחוקי לוגריתמים.

2. כעת, נגזור את הביטוי מהסעיף הקודם לפי v_x:

$$\frac{dP(D)}{dv_x} = \sum_{(x,y)\in D} v_y - \frac{\sum_{z'\in v\setminus\{x\}} v_z' * e^{v_x * v_{z'}}}{\sum_{z\in v\setminus\{x\}} e^{v_x * v_z}}$$

$$= \sum_{(x,y)\in D} (v_y - \sum_{z'\in v\setminus\{x\}} v_z{}' \frac{e^{v_x * v_z'}}{\sum_{z\in v\setminus\{x\}} e^{v_x * v_z}}) = \sum_{(x,y)\in D} (v_y - \sum_{z'\in v\setminus\{x\}} v_z' * p(z|x))$$

3.

a. במידה וישנן 500,000 מילים וכל אחת מיוצגת על ידי וקטור באורך 500, מספר הפרמטרים הינו מכפלת שני המספרים הללו: $|\theta| = 500 * 500{,}000 = 2.5 * 10^8$. נציין כי הפרמטרים של ה-*word2vec* הינם הייצוגים של כל המילים האפשריות באוצר המילים.
דרך נוספת להסתכל על כך היא בהגדרת רשת הנוירונים אשר מממשת את האלגוריתם - ארכיטקטורת *Fully Connected* אשר מחברת בין וקטור כניסה באורך מספר המילים באוצר מילים (וקטור *1-bit hot*), 500,000 במקרה שלנו, והפלט הינו באורך ה-*embedding*, 500 במקרה שלנו. כל נוירון בשכבה הכניסה מחובר לכל נוירון בשכבת הפלט ועל כן למודל יש 500,000 * 500 פרמטרים.

b. באופן כללי בלמידת מכונה ישנו כלל אצבע לפיו על כמות המידע להיות גדולה מכמות הפרמטרים. הדבר חיוני להצלחת תהליך האופטימיזציה וללמידת המודל. היות ובמודל שלנו יש סדר גודל של 10^8 פרמטרים, נזדקק לכמות דוגמאות בסדר גודל של לפחות 10^9.

c. מודל שכזה אינו מעשי מבחינה חישובית, בשל גודלו העצום. למשל, לצורך חישוב הביטוי במכנה ה-*softmax*, עלינו לחשב אקספוננט של מכפלה פנימית בין וקטורים באורך 500, חצי מיליון פעמים ולסכום את כולם. זהו כאמור רק חלק קטן מתהליך האופטימיזציה שכן יש לבצעו עבור כל אפשרות ל-v_z ולעבור על כל הזוגות האפשריים $(x,y) \in D$. לכן, מודל זה אינו מעשי חישובית.

4. להלן הביטוי אותו נתבקשנו לפתח:

$$\theta^* = argmax_\theta \sum_{(x,y)\in V^2} \log\,(p(D|x,y;\theta)$$

$$= \sum_{(x,y)\in D} \log\,(p(D=1|x,y;\theta) + \sum_{(x,y)\in V^2\setminus D} \log\,(p(D=0|x,y;\theta) =$$

$$\sum_{(x,y)\in D} log\sigma(v_x * v_y) + \sum_{(x,y)\in V^2\setminus D} \log\,(1 - \sigma\big(\mathrm{v_x} * v_y\big))$$

5. כעת נגזור את הביטוי לפי v_x:

$$\frac{dP(D)}{dv_x} = \sum_{(x,y)\in D} \frac{\left(v_y * \sigma(v_x * v_y) * \left(1-\sigma(v_x * v_y)\right)\right)}{\sigma(v_x * v_y)} - \sum_{(x,y)\in V^2\setminus D} \frac{v_y * \sigma(v_x * v_y) * \left(1-\sigma(v_x * v_y)\right)}{1-\sigma(v_x * v_y)}$$

$$\sum_{(x,y)\in D} v_y * \left(1-\sigma(v_x * v_y)\right) - \sum_{(x,y)\in V^2\setminus D} v_y * \sigma(v_x * v_y)$$

6. ישנו קושי חישובי מהותי במודל זה, היות שסיבוכיות החישוב לעיל הינה $O(|V^2|)$. גודל זה הינו עצום ולא מעשי מבחינה חישובית היות והקבוצה V הינה כל המילים בשפה.

7. לפי הסעיף הקודם, ישנו קושי חישובי בתהליך האופטימיזציה של המודל. נשים לב כי הגרדיאנט שפותח בסעיף 5 מורכב מ-2 ביטויים - הראשון סוכם על פני סט זוגות מילים עוקבות מתוך הקורפוס הנתון, בעוד שהשני עוקב אחר כל יתר זוגות המילים. גודל הסט השני גדול משמעותית מגודל הסט הראשון שכן כמות המילים השפה ענקית וכמות הזוגות שיופיעו יחדיו קטנה משמעותית מסך כל האפשרויות לצימוד מילים. על כן, הביטוי השני הינו הביטוי היקר יותר חישובית.

יתר על כן, היות והקבוצה D^c גדולה משמעותית מ- D, לביטוי השני בפונקציית ה-*loss* חשיבות גדולה יותר מהביטוי הראשון. דבר זה עלול להביא לכך שהמודל יקנה חשיבות יתרה ללמידה של ייצוגים וקטוריים רחוקים עבור מילים שאינן צמודות זו לזו ב-D, מאשר ללמידת ייצוגים דומים עבור מילים שכן הופיעו בצמידות. ניתן לחשוב על פונקציית ה-*loss* כבעלת 2 חלקים - הראשון הוא ה-*objective* - לייצג מילים דומות על ידי וקטורים דומים ואילו השני הוא הרגולריזציה - לייצג מילים שלא הופיעו בסמיכות על ידי וקטורים שונים. תחת הסתכלות זו, פונקציית ה-*loss* מקנה חשיבות יתר לרגולריזציה מאשר ל-*objective* האמיתי, כלומר, חשיבות יתר ללמידת ייצוגים שונים עבור מילים מ-D^c מאשר למידת ייצוגים דומים עבור זוגות מילים מ-D.

8. שיטת ה- *negative sampling* מסייעת הן עם הבעיה החישובית אשר הוצגה בסעיף 6, והן עם הבעיה שהוצגה בסעיף 7. במקום לחשב סכום של כל זוגות הקבוצה D^c, נחשב סכום של כל זוגות הקבוצה-D', אשר קטנה ממנה משמעותית (אנו בוחרים את גודלה). כמו כן, היות ו-D' קטנה משמעותית מD^c, המשקל של ביטוי זה בביטוי ה-$Loss$ הכולל קטן אף הוא, דבר אשר עתיד לאפשר למודל להקנות חשיבות יתרה ללמידת ייצוגים וקטוריים דומים עבור מילים שהופיעו בהקשרים דומים.

9. אנו צופים כי מילים בעלות ייצוגי *word2vec* דומים תהיינה *associated* ולא *similar* מכיוון שהביטוי אותו אנו ממזערים מתחשב בזוגות מילים עוקבות. מילים דומות במשמעות בדרך כלל לא תופענה זו בסמוך לזו אלא זו במקום זו. אי לכך, מילים כאלו יופיעו לרוב בקבוצה D' - קבוצת מילים עבורן לא נדרוש דמיון בווקטורי הייצוג. לעומת זאת , מילים שהן *associated* תופענה לעיתים קרובות בסמוך זו לזו, כלומר, בקבוצה D, דבר שיביא לכך שווקטורי הייצוג שלהן יהיו דומים.