מבואות

מידול שפה - בהינתן רצף מילים מה היא המילה הבאה: $\mathbb{P}(w_1|START)\cdot\mathbb{P}(w_2|w_1)\cdot\ldots\cdot\mathbb{P}(w_n|w_1,\ldots,w_{n-1})$

קורפרנס - "החתול נפל אבל הוא נחת על הרגליים" המילים 'החתול' ו-'הוא' מתייחסות לאותה היישות.

אנאפורה - קודם אומרים את הביטוי ואז את האזכור.

קאטפורה - קודם אומרים את האזכור ואז את הביטוי.

מרחק עריכה \ לוינשטיין -
מטריקה להשוואה בין מחרוזות:

↑	- הוספת תו
→	- מחיקת תו
↗	- יישור מחרוזות והשוואה

לבחור את המינימאלי מבין השלושה.

ביטוי רגולרי -

.	- כל תו מלבד שורה חדשה
\w \d \s	- מילה, ספרה, מרחב לבן
\W \D \S	- לא מילה, ספרה, מרחב לבן
[abc]	- אחד מם, b או c
[^abc]	- לא a, b או c
[a-g]	- תו בין a לg
^abc$	- מתחיל \ נגמר במחרוזת
\b \B	- מילה בגבולות \ לא מילה בגבולות
a* a+ a?	- 0 או יותר, 1 או יותר, 0 או 1
a{5} a{2,}	- בדיוק חמש, שתיים או יותר
a{1,3}	- בין אחד לשלוש
ab | cd	- תואם ab או cd
\\ \. \$	- הופך תו מיוחד לתו רגיל

פקודות UNIX -

`cat file1`	- כותב את כל הקובץ "file1" למסוף
`more file1`	- מציג את הקובץ "file1" עמוד בכל פעם
`less file1`	- גרסה מגוונת יותר של "more", אך פחות נפוצה
`head -30 file1`	- מציג את 30 השורות הראשונות
`tail -20 file1`	- מציג את 20 השורות האחרונות
`wc file1`	- סופר שורות, מילים ותווים בקובץ
`grep '[A-M]' file1`	- מדפיס את השורות המכילות אותיות גדולות בטווח של A עד M
`sort file1`	- ממיין את הקובץ בסדר אלפביתי.
`sort -n file1`	- ממיין את הקובץ באופן מספרי. 12 בא אחרי 2.
`sort -r file1`	- ממיין את הקובץ בסדר הפוך.
`sort -u file1`	- מסיר שורות כפולות, ומבטיח שכל שורת פלט היא ייחודית.
`uniq file1`	- מתוך שורות סמוכות אחת או יותר, מוציא רק אחת מהם.
`sort file.txt | uniq -c`	- רושם לפני כל שורה את גודל הבלוק.
`tr 'A-Z' 'a-z'`	- מחליף uppercase ל-lowercase
`tr -d ' '`	- מוחק רווחים
`tr -dc ' '`	- משאיר רק רווחים. c עבור complement
`tr -s ' '`	- מקבץ רצף תווי רווח להיות רווח אחד
`diff file1 file2`	- משווה בין שני קבצים
`tr -sc 'A-Za-z' '\n' < file1 | tr 'A-Z' 'a-z'`	- מביא תמניות tokens ללא תלות בuppercase
`[upper box] | sort | uniq`	- מביא תבניות types ללא תלות בuppercase

תמנית Token - כל המילים בטקסט כולל כפילויות.

תבנית Type - אוצר המילים בטקסט, מופע יחודי.

צורת יסוד lemma - "הלכתי" - הלך, be - "are, is".

גבעול stem - ניחוש צורת יסוד עבור שפה לא מוכרת: stori - "stories".

Type Token Ration (TTR) - $\frac{\text{תבניות}}{\text{תמניות}}$

התפלגות מילים בטקסט היא זיפפיאנית -
עבור המילה w_n ה-n'ית בשכיחותה: $\mathbb{P}(w_n)\propto\frac{1}{n_s}$

סיווג מסמכים

מטריצת בלבול-

אמיתי \ חיזוי	+	-
+	TP	FN
-	FP	TN

Accuracy - כמה המודל צודק $\frac{TP+TN}{TP+TN+FP+FN}$

Precision - כמה המודל פוגע $\frac{TP}{TP+FP}$

Recall - כמה המודל תופס $\frac{TP}{TP+FN}$

מטריקת F - ממוצע הרמוני בין P ל-R.

$$F_\beta=\frac{(\beta^2+1)PR}{\beta^2P+R}$$

כדי לחזק את R מגדילים את β.

סיווג להרבה תגים (מאקרו) - חישוב P,R,F עבור כל אחד מהתגים ואז נמצע את F. מטיב עם התגים הקטנים.

סיווג להרבה תגים (מיקרו) - נחשב P,R עבור הכל ביחד ונחשב F אחד, התגים הגדולים מקבלים יותר משקל.

פיצ'רים אפשריים במערכת סיווג: ספירת מילים, רשימות מילים המוגדרות כבעלות תכונה-ניתן לספור הופעות, מספר types,סימני פיסוק, מבנים תחביריים.

n-gram (מידול שפה) - מחשיבים רק את $n-1$ המילים שקדמו למילה ה-k'ית בשביל לחזות אותה:

$$\mathbb{P}(w_k|w_{k-n+1},\ldots,w_{k-1})=\frac{\text{count}([w_{k-n+1},\ldots,w_{k-1},w_k])}{\sum_{w'}\text{count}([w_{k-n+1},\ldots,w_{k-1},w'])}$$

חוק בייס - $\mathbb{P}(c|d)=\frac{\mathbb{P}(d|c)\cdot\mathbb{P}(c)}{\mathbb{P}(d)}$

אלגוריתם Naïve base -
קלט: מסמך d, אוסף תגים $c=\{c_1,\ldots,c_k\}$
וסט אימון $\{(d_1,c_1),\ldots,(d_n,c_n)\}$
פלט: פונקציית מסווג $\gamma:d\to c$
עבור $\mathbb{P}(c_i)=\frac{\#c_i\ in\ train}{size\ of\ train\ (n)}$
מניחים כי המסמך הוא bag of words: $d\cong\langle w_1,\ldots,w_m\rangle$
ועבור $\mathbb{P}(d_j|c_i)=\mathbb{P}(\langle w_1,\ldots,w_m\rangle|c_i)=\prod_{j=1}^{m}\mathbb{P}(w_j|c_i)$
כאשר $\mathbb{P}(w_j|c_i)=\frac{\#w_j\ in\ c_i}{\#all\ words\ in\ c_i}$
נקבל פונקציית המסווג:

$$\hat{c}=\underset{i\in\{1,\ldots,k\}}{\text{argmax}}\left(\mathbb{P}(c_i)\cdot\prod_{j=1}^{n}\mathbb{P}(w_j|c_i)\right)$$

ובלוג:

$$\hat{c}=\underset{i\in\{1,\ldots,k\}}{\text{argmax}}\left(\log\mathbb{P}(c_i)+\sum_{j=1}^{n}\log\mathbb{P}(w_j|c_i)\right)$$

אם מילה לא מופיעה באחד התגים זו בעיה, כי argmax יאופס ונכן נבצע החלקה עם 1:

$$\mathbb{P}(w_j|c_i)=\frac{1\ +\ \#w_j\ in\ c_i}{1\ +\ \#all\ words\ in\ c_i}$$

מערכת סיווג

מרכיבים -
1. ייצוג לקלט (פיצ'רים f)
2. פונקציית הסיווג מחזירה $\hat{y}$
3. פונקציית הפסד
4. אלגוריתם למידה (SGD)

סיגמויד - $\sigma(z)=\frac{1}{1+\exp(-z)}$

תכונות סיגמויד - $\sigma(-z)=1-\sigma(z)$

$$\sigma(z)'=\sigma(z)\big(1-\sigma(z)\big)$$

רגרסיה לוגיסטית - פונקציית סיווג $\sigma(\theta\cdot f+b)=\hat{y}$

אלגוריתם SGD -
נחשב $\hat{y}^{(t)}=\sigma\left(f\cdot\theta^{(t-1)}\right)$ ונעדכן η אם צריך.
פונקציית ההפסד:

$$L_{CE}=-\log\left(\mathbb{P}(y|x)\right)=$$
$$=-\left[y\cdot\log\left(\sigma(\theta\cdot f+b)\right)+(1-y)\cdot\log\left(1-\sigma(\theta\cdot f+b)\right)\right]$$

נגזור פונקציית הפסד: $\nabla L_{CE}\left(\theta^{(t-1)}\right)=\left(\hat{y}^{(t)}-y\right)\cdot f$
ונוכל לעדכן: $\theta^{t+1}=\theta^t-\eta\cdot\nabla L_{CE}\left(\theta^{(t-1)}\right)$

רגולריזציה - לבלום במפורש את ערכי הפרמטרים וגם להתמודד עם overfitting: $L=L_{CE}+L_{reg}$

L2 - $L_{reg}=\|\theta\|_2^2$ מענישה מודלים שהתרחקו מהראשית.

L1 - $L_{reg}=\|\theta\|_1$ מענישה מודלים שיותר מדי פרמטרים שלהם אינם 0.

מודל overfitting - התאים את עצמו יותר מידי למדגם אימון (או שפשוט אין בו מספיק פרמטרים)

רגרסיה מרובת קלאסים - נחשב את הציון לכל קלאס ובסוף נגרמל בסכום כל הציונים. נחזיק גם וקטור משקולות עבור כל קלאס. פונקציית ההפסד:

$$L_{CE}=-\log\left(\frac{\exp(w_k\cdot x+b_k)}{\sum_{j=1}^{K}\exp(w_j\cdot x+b_j)}\right)$$

נגזור פונקציית הפסד: $\frac{\partial L_{CE}}{\partial w_{k,i}}=-\left(\frac{\exp(w_k\cdot x+b_k)}{\sum_{j=1}^{K}\exp(w_j\cdot x+b_j)}\right)\cdot x_i$

כל הפרמטרים מתעדכנים שכן softmax אף פעם לא מחזיר 0 או 1.

פרספטרון - נכנה לייצר קו מבחין בין דוגמאות חיוביות ושליליות, אם צדקנו אין עדכון אחרת נעדכן לפי הקלאס הנכון.

תיוג רצפים

בעיות - תיוג חלקי דיבר, תיוג תכונות מורפותחביריות, זיהוי יישויות.

תכונות מורפותחביריות - מין, גיל, זמן, בניין, גוף, יחיד\רבים, voice (active/passive).

חלקי דיבר -
open class:
שם תואר adj, תואר הפועל adv, שם עצם noun, פועל verb, שם פרטי propn, קריאות itnj (היי, יו),
closed class:
מילות יחס adp, פעלי עזר aux, מילים שמחברות בין חלקי משפט שווים cconj, מילות חיבור משעבדות (כלומר מחברות בין משפט פחות חשוב לאחד יותר חשוב) sconj, כינויי גוף pron, מה שיש רק באנגלית particles (עם wake), מספרים num, מילת יחס prep.
other class:
סמלים sym, כל השאר x.

PTB - הקורפוס המסיבי הראשון הגדול באנגלית.

מטריקות תיוג רצפים:
- דיוק ברמת המילה (accuracy)
- F_1 על התגים (מאקרו\מיקרו)

מטריקות זיהוי יישויות:
- טעינו איפשהו ביישות – טעות בכולה
- חישוב F_1 לכל היישויות או לאחת

סכימת BIO - B תחילת ישות, I אמצע, O לא חלק.

שרשרת מרקוב - מניחים כי כל מצב תלוי רק במצב שקדם לו: $\mathbb{P}(q_i|q_1,\ldots,q_{i-1})=\mathbb{P}(q_i|q_{i-1})$
נצטרך גם את ההסתברות להיות במצב הראשון π.

מודל מרקוב נסתר HMM – מודל גנרטיבי, ראים רק את המילה (הפלט) ואנחנו רוצים להבין איזה מצב (חלק דיבר) יצר אותה. המצבים הם ה"נסתרים". מניחים כי פלט תלוי רק במצב שלו.

ההסתברות של רצף התגים בהינתן רצף המילים:

$$\hat{t}_{1:n}=\text{argmax}\,\mathbb{P}(t_1,\ldots,t_n|w_1,\ldots,w_n)$$
$$\approx\text{argmax}\prod_{i=1}^{n}\mathbb{P}(w_i|t_i)\mathbb{P}(t_i|t_{i-1})$$

אלגוריתם ויטרבי - תכנון דינמי, נרצה למצוא את המסלול הכי סביר בין המצבים שייצרו את התצפיות (המילים במסמך).
קלט:

$Q=q_1,\ldots,q_N$	קבוצת N המצבים (מיקום המילה במשפט)
$A=a_{11},\ldots,a_{NN}$	מטריצת מעברים כך שa_{ij} שווה להסתברות לעבור ממצב i למצב j
$O=o_1,\ldots,o_T$	קבוצת T התצפיות (מילים) מתוך אוצר המילים
$B=b_i(o_t)$	סיכוי קבלת תצפית o_t במצב q_i
$\pi=\pi_1,\ldots,\pi_N$	התפלגות התחלתית מעל המצבים

ונוכל למצוא:

$$v_t(j)=\max_{q_1,\ldots,q_N}\mathbb{P}(q_1,\ldots,q_{t-1},o_1,\ldots,o_t,q_t=j|\lambda)$$
$$=\max_i v_{t-1}(i)\cdot a_{ij}\cdot b_j(o_t)$$

ונעבוד בלוג:

$$v_t(j)=\max_i\log v_{t-1}(i)+\log a_{ij}+\log b_j(o_t)$$

דוגמא -

$$O=\{o_1=\text{שרה},o_2=\text{שיר},o_3=\text{שמח}\}$$
$$Q=\{q_1=V,q_2=N,q_3=M,q_4=P,q_5=J\}$$

טבלאת $B=Emissions(\log)$:

	שרה	שיר	שמח
V	-3.912	-6.215	-6.908
N	-5.298	-4.605	0
M	-3.507	-3.912	-6.908
P	0	-11.513	0
J	0	0	-3.219

טבלאת $A=Transitions(\log)$:

→	V	N	M	P	J
V	-2.996	-1.050	-1.386	-1.609	-1.897
N	-0.511	-2.996	-2.996	-2.303	-1.609
M	-0.511	-2.303	-2.303	-1.897	-2.996
P	-2.996	-0.693	-1.204	-4.605	-1.966
J	-0.693	-2.996	-2.303	-1.605	-1.897

התפלגות התחלתית π:

V	N	M	P	J
-1.897	-1.386	-1.386	-1.204	-2.996

בשורה עבור "שמח" נפעל בדומה, ונקבל כי הערך המקסימלי $v_4(5)=-19.799$ אשר התקבל ממצב N בתוך תא 2. לכן, תיוג האלגוריתם יהיה שמח=J, שיר=N, שרה=V, שרה=M.

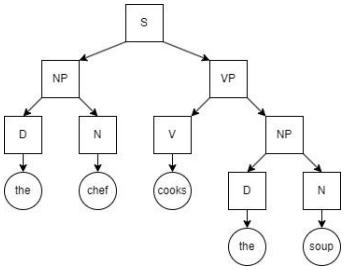
CRF - מודל דיסקרמינטבי. כל תג תלוי בתג הקודם במיקום עצמו ובכל הקלט (לצורך פיצ'רים)

מידול מבנים ותלויות (תחביר)

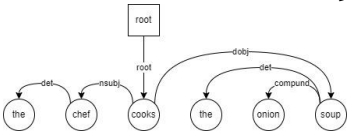
דקדוק חסר הקשר CFG - מוגדר על ידי:

N	קבוצת הסמלים הלא סופיים (צירופים\תגים)
Σ	קבוצת הסמלים הסופיים (מילים)
R	כללי גזירה מהצורה $A\to B$ כאשר A לא סופי
S	סמל התחלתי וחלק מN

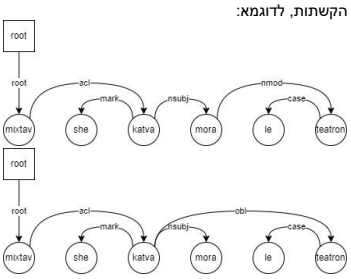
חומסקי CNF - כל כלל כולל מימין שני סמלים לא סופיים או מילה אחת סופית.



עצי תלויות -



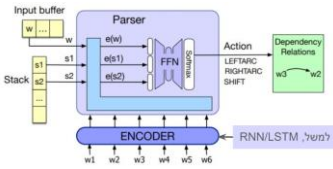
ניתן לבטא כמה משמעויות דקדוקיות שונות בעזרת שינוי הקשתות, לדוגמא:



הטלויות של עץ תלויות - תכונה לפיה קשתות העץ לא חוצות זו את זו.

סוגי קשתות בעץ תלויות -
קשתות בין נשוא (פרדיקט) לארגומנט ליבה:
- נושא שמני nsbdj
- נושא פסוקי csubdj
("ללמוד הרבה זה חשוב" ⟵ "ללמוד הרבה")
- מושא ישיר dobj
- מושא עקיף iobj
("he gave her the book" ⟵ "her")
- פסוקית משלימה ccomp\xcomp
("הוא אמר לה לסדר את החדר" ⟵ "לסדר את החדר")
קשתות בין נשוא למרכיבי עזר:
- נספחים שמניים obl
("הוא הלך הביתה אתמול" ⟵ המילה אתמול היא לא מרכיב ליבה הכרחי של המילה הלך, כלומר גם בלעדיה המפשט תקין)
- פסוקית אופן advcl (במהירות, ביעילות)
- פועל עזר cop, aux
קשתות בתוך צירופים שמניים:
- תוויות הידוע determiner כמו the
סמיכות nmod כמו "כיתת לימוד"
מילת יחס case

ניתוח תלויות במעברים - החלטות מקומיות, 3 מבנים:
- buffer: מחזיק את כל המילים שטרם טיפלנו
- stack: מועמדים לקבלת קשת
- list of edges: הקשתות שהוספנו לעץ



מעברים -
- shift: העברת המילה מהbuffer לstack
- left-arc: הוספת קשת מהמילה העליונה במחסנית לזו שמתחתיה ונוציא את השנייה
- right-arc: הוספת קשת מהמילה השנייה לעליונה והוצאת העליונה (אי אפשר לקבלת קשתות חוצות, רק הטליות)

parser - לומד איזו החלטה לבצע בהינתן:
1. מצב של המערכת (חיץ, מחסנית, קשתות קיימות)
2. החלטה מועמדת (הזה, קשת-ימין, קשת-שמאל)

eager - יוצרים קשתות בין ראש stack לתחילת הbuffer. כאשר מיצרים קשתות ימינה ברגע שאפשר בלי להיפטר מהמילה העליונה (עם הגדרת פעולה חדשה להוציא מראש המחסנית reduce).

oracle -
קלט: עץ מוכן, אוסף קשתות לא סדור.
פלט: סדרת החלטות שהובילה ליצירת העץ מהקלט.
נתינת הציונים למעברים היא בעיית סיווג שניתן לבצע ע"י חילוץ פיצ'רים מתוך המערכת ואז לתת ציון לכל פעולה.

search Beam - מחזיקים את K תתי העצים הכי טובים עד כה, בכל צעד מתייחסים לכל העלומה ומעדכנים את מה ששאר בה.

ניתוח תלויות בגרף - בעל תוצאות טובות יותר מאשר ניתוח תלויות במעברים. נעבור על כל העצים האפשריים וניקח את העץ בעל הציון המקסימלי (סכום הקשתות). נסתכל על עץ שלם וניתן ציון כולל. מתבסס על ההנחה שציון קשת לא תלוי בקשתות האחרות. נגדיר ציון לכל קשת, לכל זוג צמתים, לכל הכיוונים, ולכל סוג קשת אפשרית וגם לכל צומת כשורש. (עולה $O(n^3)$)

מציאת עץ פורש מקסימלי- ניתן ציון לכל הקשתות, נמצא לכל צומת את הקשת המקסימלית הנכנסת אליו. אם יש מעגל:
1. נפחית מכל קשת הנכנסת לצומת את הערך המקסימלי שנכנס לצומת
2. נכווץ את המעגל לצומת בודד ונקרא לאלגוריתם על העץ החדש
3. נרחיב מחדש את העץ ונבחר את הקשת לפי העץ המוקטן

ציון התאמה ללא תווית (UAS) - אחוז המילים שקיבלו את הראש הנכון.

ציון התאמה עם תווית (LAS) - אחוז המילים שהוקצו להן הראש הנכון ותג התלות הנכון (או "תווית"). לכן מחמיר יותר מUAS ומתקיים LAS<UAS.

שיכוני מילים (word embeddings)

שיכוני מילים - וקטור השיכונים של מילה מסוימת במרחב הוקטורי.

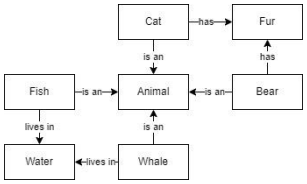
פוליסמיה - מילה אחת עם כמה משמעויות (עכבר). מילה אחת, הקשרים שונים. וקטור ממוצע של וקטורים רחוקים.

מילים נרדפות - כמה מילים עם אותה משמעות (ירח\לבנה). כמה מילים, אותו הקשר. וקטורים קרובים.

מילים דומות - מתארות חפצים או רעיונות דומים בעולם (כלב\חתול). כמה מילים, הקשרים דומים. וקטורים קרובים עם קשר אלגברי.

מילים קשורות - מתארות חפצים או רעיונות הקשורים לאותו שדה סמנטי (כלב\מלונה). מילים בהקשר זו של זו. וקטורים עם קואורדינטות מסוימות דומות.

רשתות סמנטיות - מילים (lemmas) ממופות למובנים (senses) והמובנים מקושרים ביניהם בצורת גרף.



האם אוסף השיכונים באמת מייצג יחס דמיון לשוני אמין?

1. ניתן לשאול אנשים עד כמה המילים דומות. זה סובייקטיבי.
2. נשתמש בדמיון קוסינוס:

$$cosine(v,w) = \frac{v \cdot w}{\|v\|_2 \|w\|_2}$$

שלילי עבור skip-gram ו- PMI.

מטריצת מילים\מסמכים - נייצג כל מילה לפי אוסף המסמכים שהיא מופיע בו וכמה בכל מסמך.
חסרונות:

1. וקטורים ארוכים לכל מילה.
2. מילה כמו the תהיה דומה לשאר המילים functionable ויהיה קשה ללמוד עליהן משהו.

מטריצת שכנויות - נספור הופעות של זוגות מילים באותו מסמך. w היא המילה שמעניינת אותנו, ו- c הוא ההקשר.

PMI - זוהי נוסחא שעוזרת לנו להתגבר על כך שחלק מהמילים שכיחות יותר וחלק מהמילים פחות בכך שאנחנו מנרמלים את ההסתברויות:

$$\mathrm{PMI}(w,c) = \log_2 \frac{\mathbb{P}(w,c)}{\mathbb{P}(w)\cdot\mathbb{P}(c)}$$

כאשר:
$\mathbb{P}(w,c)$: ההסתברות ש-w תופיע במסמך עם הקשר c
$\mathbb{P}(w)$: ההסתברות ש-w תופיע במסמך
$\mathbb{P}(c)$: ההסתברות ש-c תופיע במסמך
כשהתוצאה תהיה שווה לאפס זה אומר שיש אי-תלות בין המילים.

PPMI - ביצוע RELU על PMI (מבטיח רק חיוביות)

TF-IDF - גישה מקובלת באחזור מידע, מחפשים שאילתה במנוע חיפוש ורוצים לדעת איזה מסמך להחזיר:

$$\mathrm{TF\_IDF}_1(t,d) = \log(1+f_{t,d}) - \log\left(\frac{N}{n_t}\right)$$

כאשר:
- $f_{t,d}$: כמות הפעמים שהמילה t מילה מתוך השאילתה הופיעה במסמך d
- n_t: כמות המסמכים שהמילה t הופיעה בהם
- N: כמות המסמכים

החיסור נועד לתת משמעות למילים נדירות

שיכון ע"י חיזוי הקשרים מקומי (skip-gram) -
נפעיל אלגוריתם חיזוי, אבל נשמור רק את המשקולות הנלמדים.
האלגוריתם:
בהינתן מילה w, נסה לחזות את השכנות שלה c בהסתברות גבוהה ככל הניתן, ונעבור כך על כל המילים בקורפוס. בעצם נחשב מה ההסתברות w,c מופיעות ביחד $\mathbb{P}(+|w,c)$.
(לכל מילה יהיה וקטור מטרה בתפקידה כחוזה, ווקטור הקשר בתפקידה כנחזית.

היפר פרמטרים -
- רוחב החלון (בכל פעם נדגום רק "חלון" בסביבת המילה וכך נגדיר מה יהיו מילות ההקשר)
- כמות הדוגמאות השליליות לכל דוגמה חיובית
- מימד וקטור השיכון
- α עבור דגימת המילים השליליות מתוך התפלגות שמוסחת לטובת מילים נדירות:

$$\mathbb{P}_\alpha(c) = \frac{(\#c)^\alpha}{\sum_{c'}(\#c')^\alpha}$$

$\alpha = 0.75$ זה ערך טוב.

יעד החיזוי - נשתמש ברגרסיה לוגיסטית על מנת לחשב את ההסתברות של $\mathbb{P}(c|w)$ כאשר על כל דוגמה חיובית ניקח כמה שליליות:

$$\mathbb{P}(+|w,c_{pos}) \cdot \prod \mathbb{P}(-|w,c_{neg})$$

פונקציית ההפסד:

$$L_{CE} = -\left[\log\left(\sigma(c_{pos}\cdot w)\right) + \sum_{i=1}^{k}\log\left(\sigma(-c_{neg}\cdot w)\right)\right]$$

נגזור פונקציית הפסד:

$$\frac{\partial L_{ce}}{\partial c_{pos}} = \left[\sigma(c_{pos}\cdot w) - 1\right]\cdot w$$

$$\frac{\partial L_{ce}}{\partial c_{neg}} = \left[\sigma(c_{neg}\cdot w)\right]\cdot w$$

$$\nabla L_{CE} = \left[\sigma(c_{pos}\cdot w) - 1\right]\cdot c_{pos} + \sum_{i=1}^{k}\left[\sigma\left(c_{neg_i}\cdot w\right)\right]\cdot c_{neg_i}$$

ומוכל לעדכן: $w^{(t+1)} = w^{(t)} - \eta\cdot\nabla L_{CE}\left(w^{(t)}\right)$

מטריקות לשערוך מודלי דמיון -
קלט: שתי רשימות מדורגות של הזוגות הנשפטים.
פלט: מספר בטווח $[-1,1]$ שמתאר את הקשר שבין הרשימות

- pearson correlation

$$\rho(X,Y) = \frac{\mathrm{cov}(X,Y)}{\sigma(X)\cdot\sigma(Y)}$$

אפשר לכווץ ולהרחיב את התחום. רגיש לפיזור הערכים בשונים בתוך של X.

- spearman correlation

$$r_s = \rho\left(R(X),R(Y)\right) = \frac{\mathrm{cov}\left(R(X),R(Y)\right)}{\sigma\left(R(X)\right)\cdot\sigma\left(R(Y)\right)}$$

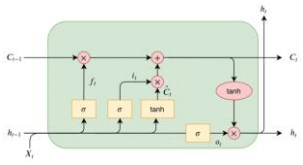
עבור R פונקציית דירוג שמעניינת את הציונים של כל אחת מהרשימות.

רשת ניורונים

רשת בהיזן קדמי FFN - המון שכבות של נוירונים פשוטים בתוספת פונקציית אקטיבציה ביניהם.
פרספטרון רב שכבות MLP - אותו דבר רק שהנוירון הפשוט הוא פרספטרון.
אקטיבציות אפשריות -

1. RELU - בחירה טובה, גזירה קלה
2. סיגמויד - ערכים בין 0 ל-1
3. tanh - ערכים בין 1- ל1

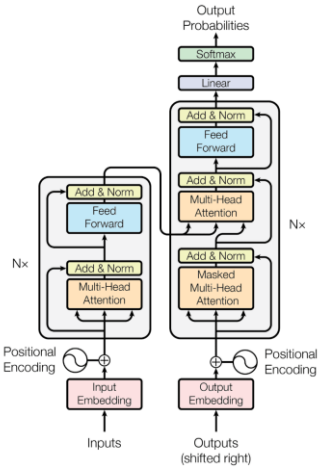
אימון הרשתות - מציאת הנגזרות החלקיות לפי הlossו ומפעפעים אחורה (back propagation או כלל השרשרת).
מגבלות הFFN - התייחסות לכל הקלט בבת אחת, אין חשיבות לסדר קלט, מוגבל לחיזוי תג אחד.
רשת נוירונים נשנית RNN - בFFN לא עובר מידע בין שכבות עוקבות, ואילו בRNN כן.
LSTM - גרסה משופרת לRNN. מכיל מרכיבים שנועדו לשמור על מצבים רחוקים (long short-term memory).



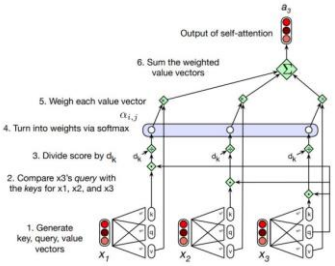
GRU - פשרה בין RNN ל-LSTM. מהיר בהרבה מ-LSTM.
Bidirectional RNN - יודע להבדיל בין כיוונים. לכל כיוון יש את הפרמטרים שלו וה"חיבור" נעשה ברמה הבאה.
רגולריזצית dropout - בכל שלב אימון נאפס חלקים אקראיים בשכבה ונתעלם מהם.
רשת אוטורגרסיבית - רשת שנעה קדימה, כלומר לומדת רק מקלטי העבר ואינה יכולה להסתכל לעתיד.

רובוטריקים

מבנה הרובוטריק -



- self-attention



- softmax

$$\mathrm{softmax}(z_i) = \frac{\exp(z_i)}{\sum_{j=1}^{k}\exp(z_j)}, \qquad i \in [1,\dots,k]$$

השלבים בself-attention -

1. השוואה בין הווקטורים בקלט ומתן ציון לדמיון שביניהם.
2. נרמול הציונים שקיבלנו ע"י שימוש בsoftmax.
3. חישוב הפלט המנחי

כל מילה שואלת כמה היא צריכה לקחת מכל מילה ברמת הייצוג הבאה.
- Query: המילה המתשאלת
- Key: המילה כגורם השוואה
- Value: המילה כמרכיב של שכבה הבאה

$$\mathrm{SelfAttention}(Q,K,V) = \mathrm{softmax}\left(\frac{Q\cdot K^T}{\sqrt{d_k}}\right)\cdot V$$

כאשר:

$$Q = X\cdot W_q, \qquad \dim(W_q) = d\times d_k$$
$$Q = X\cdot W_k, \qquad \dim(W_k) = d\times d_k$$
$$Q = X\cdot W_v, \qquad \dim(W_v) = d\times d$$

לדוגמא: בהינתן 3 מילים x_1,x_2,x_3 ואנו מעוניינים ב-y_1,y_2,y_3 בהתאמה, נקבל לדוגמא עבור y_3:

$$y_3 = \sum_{i=1}^{3} q(x_3)\cdot k(x_i)^T\cdot v(x_i)$$

הקשרים השיוריים (residual connection) - נותנים לשכבות העליונות גישה ישירה למידע המגיע מהשכבות הנמוכות.

$$z = \mathrm{LayerNorm}(x + \mathrm{MultiAttention}(x))$$
$$y = \mathrm{LayerNorm}(z + \mathrm{FFNN}(z))$$

שכבת הנורמליזציה - נחשב שונות ותוחלת:

$$\mu = \frac{1}{d_h}\sum_{i=1}^{d_h} x_i, \qquad \sigma = \sqrt{\frac{1}{d_h}\cdot\sum_{i=1}^{d_h}(x_i-\mu)^2}$$

ואז ננרמל: $\hat{x} = \frac{(x-\mu)}{\sigma}$

צומי מרובה ראשים (multi-head attention) - כמה שכבות של self-attention שנקראות ראשים, לכל ראש יש סט מטריצות משלו. כל ראש יכול למצוא קשתות שונות בין המילים במשפט.
(מוסיפים לשכבת הצומי כקלט מיקומי המילים בעזרת positional embeddings).

למידה בהעברה (transfer learning) - למידת מערכת למטרה מסוימת ושימוש באלמנטים הנלמדים שלה למשימה אחרת (כמו בskip-gram).

fine tuning - עדכון הפרמטרים שנלמדו למשימה המקורית על גבי המשימה החדשה.

מקודד (encoder) - לוקח טקסט (או אובייקט אחר) ומחזיר וקטור

מפענח (decoder) - לוקח וקטור (או כמה) ומחזיר טקסט

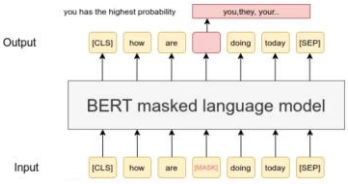
מקודד רובוטריק - כמו קודם אבל יכול לראות את כל הקלט

מפענח רובוטריק - יכול להסתכל רק אחורה בפלט, אבל יכול להסתכל על כל הקלט בעזרת cross-attention.
הלמידה כאן באמצעות קרוס אנטרופי רגיל פחות log הסתברות של המילה. נכריח את המודל בכל שלב לקחת את המילה הבאה האמיתית / חיפוש אלומה.

שני סוגים של מודלים ברובוטריק -

1. מודל שפה: מסתכל רק אחורה (autoregressive) ללמידה מעוברת
2. מודל חיזוי או מקודד לבעיות תרגום או ממוסך (מסתכל קדימה)

מודל שפה דו כיווני ELMO - חיבור של מודל קדימה ומודל אחורה.
BERT - רובוטריק כמודל שפה דו כיווני. נסתיר לחלק מהמילים את עצמן. אחוז ההסתרות המקובל הוא 15%. יכול לשמש כמקודד תרגום.



הנדסת פרומפטים

למידה בתוך הפרומפט (ICL) - הפרומפט עצמו מכיל דוגמאות, והמודל משלים על-סמך מה שראה בהן.
לדוגמא: "בהינתן טקסט ביקורת של לקוח, אני רוצה שתגיד לי האם הוא נהנה. לדוגמא: ביקורת: [ביקורת1], הלקוח [כן\לא] נהנה".
ההקשר לבדו אמור לתת למודל את היכולת לייצר הכללה. ICL צריך להיות חסכוני ולכן יעיל, כדי שכל הדוגמאות יכנסו בתוך חלון ההקשר. ICL עשוי להיות איטי, כיוון שצריך לתת מחדש את כל הדוגמאות בתוך ההקשר לכל טקסט שאנחנו רוצים לתייג, בנפרד.

שרשור מחשבה (chain of thought) -
במקום לבקש פתרון ישיר לשאלה, ננחה את המודל לענות "צעד-צעד", עובד מצוין בבעיות חשבון (ומשם התפתח). עובד בעיקר במודלים בקנה מידה גדול (100 מיליארד פרמטרים+).

אתיקה והוגנות

השפעה (influence) - X התחיל להשתמש הרבה במילה, הרשת החברתית הקרובה ל-X ממאצת אותה.
הומופיליה (homophily) - הרבה משתמשים מכירים מילה מסוימת ומתחברים לקהילה קרובה דרך השימוש המשותף.

דמיון מבני (structural equivalence) - צמתים שה"תפקידים" שלהם בתוך הרשת דומים זה לזה (אבל לא בהכרח קשורים).

הוגנות אלגוריתמית - מניחים שניתן לחלק את האוכלוסייה לפי תכונה A, שאינה רלוונטית למשימה הנלמדת (מונח מקובל הוא קבוצה מוגנת).

שונות בתוצאה - התפלגות החיזוי $\hat{y}$ בהינתן A שונה מההתפלגות של המידע האמיתי בהינתן A.

שונות בשגיאה - התפלגות שגיאת החיזוי ϵ (לתוצאה מספרית, לתג סיווג מסוים, וכו') שונה עבור A מאשר עבור לא-A.

הוגנות קבוצתית - שגיאות צריכות להתפלג באופן דומה עבור קבוצות שונות.

הוגנות פרטית - פרטים בעלי תכונות (רלוונטיות) דומות צריכים לקבל יחס דומה ללא תלות בהשתייכותם לקבוצה מסוימת.

שוויון בהדדמנויות - עבור קבוצות שונות, יחס ה"קבלה" דומה.

שוויון ביחסים - עבור קבוצות שונות, יחס השגיאות החיוביות דומה (כלומר, ההסתברויות של מועמד ראוי להתקבל ושל מועמד לא ראוי להתקבל דומים על-פני הקבוצות).

קיום התנאים במקביל - ניתן להוכיח שלא ניתן לקיים את כל התנאים האלה ביחד. מצד שני, יש גם פתח לפתרון (חלקי) - הוספת ההוגנות הפרטית כ-loss נוסף.

חילול טקסט

קלט - רצף מילים או משהו שכולל רצף מילים כמו תמונות או אודיו.
פלט - רצף מילים כלשהו.
קשיים במעבר בין שפות -

1. סדר המילים שונה
2. אי תאימות במשמעות
3. ביטוי בהטיה מורפולוגית (זמן וגוף) לעומת מילות עזר
4. אלמנטים חסרים (כינוי גוף)

משולש ווקווא -
text → syntax → semantics → interlingua

לוג'יטים -
פלט השכבה האחרונה במודל. ברובוטריק המימוש יהיה להכפיל-פנימית את ה-h שהגיע מהשכבה האחרונה בכל אחד מהשיכונים של המילים באוצר המילים: $u = h_N^L E^T$.

שיטות חילול -
- חילול חמדני: ביצוע argmax. בעייתי מכיוון שהוא צפוי, חוזר על עצמו ואפילו דטרמיניסטי.
- דגימה טהורה: דגימה בהתפלגות המוגדרת לפי וקטור softmaxה שמתקבל על הלוג'יטים. בעייתי כי קיים זנב ארוך של אוצר-הטוקנים. ההסתברות המצטברת של הזנב גבוהה ואנחנו צפויים להיתקל הרבה מאוד בטוקנים נדירים, גם אם כל אחת בנפרד קטנה מאוד.
- דגימת ראש: בהינתן k ידוע מראש, ניקח את k המילים שבראש רשימת ההסתברויות הממוינת ונזרוק את היתר. ננרמל את ההתפלגות הנותרת, כלומר נחלק את ההסתברויות המקוריות של k המילים בסכום הסתברויות שלהן.
- דגימת גרעין: נבחר סף p, ונגריל מאוסף המילים אשר סכום ההסתברויות שלהם עבר את הסף הזה. יש צורך לנרמל לאחר בחירת המילים.
- טמפרטורה: מחלקים את ערכי הלוג'יטים בסקלאר τ (היפר-פרמטר) לפני הכניסה לsoftmaxה. הנוסחא המעודכנת לשלב בניית ההתפלגות היא:

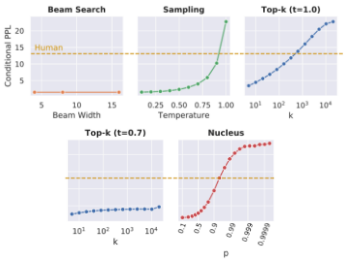
$$y = \mathrm{softmax}\left(\frac{u}{\tau}\right)$$

ברוב המקרים $\tau \in (0,1]$ כי תוספת חום מקרבת אותנו להתפלגות אחידה וקירור מקרב להתפלגות דטרמיניסטית.

ציון perplexity - כמה המודל מופתע מכל מילה

$$\mathrm{PPL}(x) = \exp\left(-\frac{1}{N}\cdot\sum_{i}^{N}\log\mathbb{P}(x_i|x_{<i})\right)$$

כאשר ערך הPPL נע בתחום $[1,\infty)$



מילים לא מוכרות (OOV)

7 סוגים של מילים כאלו -
- ישויות חדשות
- מילה מאוד מאוד נדירה
- תאריכים ומספרים
- מילים מדומיינות מיוחדות
- תחדיש
- מילים משפות זרות, מילים שאולות
- שגאיות דפסם

4 דרכים לפתרון -
- הכנסה אקטיבית של שגיאות לקורפוס אימון
- תיקון שגיאות ב-inference
- בחירת קורפוס אימון מגוון
- מידול תווים / תת-מילים

character LSTM - נועד לטפל בOOV בשלה האימון. המילה החסרה באימון תקבל את הוקטור עבור <unk>, לאחר מכן נוכל לעשות LSTM דו כיווני לחבר את התוצאות לFFN ולקבל תג מתאים לכל מילה.

fast text - גם מטפל ב OOV באימון. מתפקד כמו skip-gram אבל הsubwords הם כל רצפי התווים באורך 3-6 של המילה הרצויה עם חיפופת שמשורשר בהתחלה ובסוף המילה תגים מיוחדים $\langle s\rangle, \langle\backslash s\rangle$:

$$\mathrm{subwords}(\langle s\rangle \mathrm{child} \langle\backslash s\rangle) = \left\{\begin{array}{c} \langle s\rangle ch, chi, hil, ild, ld\langle\backslash s\rangle, \\ \underline{3-gram} \\ \langle s\rangle chi, chil, hild, ild\langle\backslash s\rangle, \\ \underline{4-gram} \\ \langle s\rangle child, child, hild\langle\backslash s\rangle, \\ \underline{5-gram} \\ \langle s\rangle child, child\langle\backslash s\rangle, \\ \underline{6-gram} \end{array}\right\}$$

ובעזרת המילים האלו:

$$\mathrm{emb}(child) = \mathrm{emb}_T(child) + \sum_{s\in\mathrm{subwords}(\langle s\rangle \mathrm{child}\langle\backslash s\rangle)} s$$

subwords tokenization - נחלק את המילה לחלקים ללא חפיפה (רוב הטרנספורמרים משתמשים בזה).