עיבוד שפה טבעית - תרגיל בית 2

תיאור המשימה

בתרגיל בית זה תממשו tokenizer על פי האלגוריתם BPE - Byte Pair Encoding כפי שנלמד בכיתה. אתם תבנו את הטוקנייזר עבור שני דאטה סטס שונים הנתונים לכם , המגיעים מדומיינים שונים - ותבחנו גם על דומיין שלישי הנסתר מכם. בנוסף, יסופק לכם גם הקוד עבור מודל (ותהליך האימון שלו) הפותר את המשימה של זיהוי ישויות בטקסט (Named Entity Recognition), אשר ישתמש בtokenizer אותו בניתם. כך תוכלו להעריך את השפעת איכות הtokenizer שלכם על משימות עיבוד שפה מעולם נתונים אמיתי. ציונכם יקבע ע״י יעילות הטוקנייזר (מספר טוקנים ומהירות ההסקה - encoding על test set נסתר) והביצועים על המשימה.

את התרגיל יש לבצע בשפת python3.

בתרגיל תדרשו לממש ולאמן שלושה טוקנייזרים – אחד עבור כל דומיין.

לאורך התרגיל הכוונה במדד F1 היא למדד F1 בינארי ברמת המילה. כלומר, שרשור כל הפרדיקציות של המודל שלכם והתיוגים האמיתיים על פני כל המשפטים, וחישוב binary F1 בין שתי הרשימות. הסבר מפורט על המקרה של sub tokens ניתן למצוא בהמשך התרגיל

לאורך התרגיל הכוונה ביעילות הטוקנייזר היא למספר הtokens שכל משפט מקודד אליהם והזמן הלוקח לו לבצע הסקה בסביבת המכונה הוירטואלית - כפי שמחושב ומודפס ע״י הפונקציה שסופקה לכם.

הסבר על מבנה הציון בתרגיל:

- **45%** - מימוש מלא של שלושת הטוקנייזרים, ועמידה ברף של ציון F1 של לפחות 0.5 על קובץ ה development עבור domain_1 וdomain_2
- **40%** - תחרות משוקללת על יעילות הטוקנייזר, מהירות הטוקנייז וציון F1 בתיוג קבצתי התחרות, בכל אחד משלושת הדומיינים
- **5%** -הערכה של הביצועים של הטוקנייזר השלישי
- **10%** - כתיבת דו"ח תמציתי (עד עמוד אחד בפונט אריאל בגודל לפחות 10 עם שוליים סטנדרטיים אשר יכלול את הסעיפים הנדרשים ועמידה בתנאי פורמט ההגשה (יפורטו בהמשך המסמך). דו״ח שלא יעמוד בדרישות ההגשה יקבל0/10.

נתונים:

קבצי הנתונים של התרגיל הם בפורמט הבא:

1. שני קבצי האימון לטוקנייזר - domain_1_train.txt, domain_2_train.txt - כל משפט מופרד בירידת שורה. אתם יכולים לבחור להשתמש בהם כרצונכם. כמו כן קבצי dev בהתאמה. שימו לב שאתם לא מקבלים קובץ אימון עבור הדומיין השלישי והנסתר, ועליכם להשתמש בדאטה שניתן לכם בלבד.
2. הקבצים הנדרשים לאימון והערכת ביצועים על המשימה -
    a. קובץ האימון הוא הקובץ train_1.tagged, קובץ ה development הוא dev_1.tagged עבור הtokenizer מהdomain הראשון.

b. קובץ האימון הוא הקובץ train_2.tagged, קובץ ה development הוא dev_2.tagged עבור הtokenizer מהdomain השני.

אינכם נדרשים לקרוא ולעבד קבצים אלו בעצמכם, אלא לוודא שבקובץ train_ner_model.py אתם מעבירים את הפרטמטרים הנכונים בהתאם לdomain. שימו לב כי מלבד ניתוב הקבצים (אימון, dev, והנתיב לטוקנייזר שלכם, אין לשנות אף פרמטר של ההרצה - הטוקנייזר שלכם יבחן אל מול מודל אחיד שהתאמן באותו אופן אצל כולם. הסבר נוסף על כיצד להשתמש בקובץ ובקבצים הנוספים נמצא בקובץ הreadme.

אימון

מימוש כל הטוקנייזרים שלכם חייב להתבסס על האלגוריתם - BPE - Byte Pair Encoding - כפי שנלמד בכיתה. כלומר - הוא מתחיל מפירוק המשפט לרמת התו.

בנוסף, הטוקנייזר חייב לאפשר טוקנים עד רמת הbigram - כלומר שני מילים סמוכות המקבלות token אחד. עליכם לוודא שכל טוקנייזר יצר לפחות bigram token אחד - <u>אי עמידה בתנאי זה תגרור לפסילת התרגיל!</u>

<u>בדו"ח התרגיל</u> דווחו על הbigrams הנפוצים ביותר (עד 5) והכי פחות נפוצים (עד 5) ושחיכותם בסט האימון.

אנחנו מספקים לכם מחלקה הבסיסית BaseTokenizer - על כל אחד מהטוקנייזרים שלכם לרשת ממנה (:class YourTokenizer(BaseTokenizer) - אתם רשאים להוסיף במחלקה שלכם פונקציות ומשתנים כרצונכם, אך שימו לב שאתם ממשים <u>במחלקה שלכם</u> את כל הפונקציות הנדרשות ע"י המחלקה האבסטרקטית. הקובץ train_tokenizer.py נועד כדי לעזור לכם בתהליך הרצת האימון. שימו לב שאתם משנים את החלקים בו כתוב BPETokenizer (שאינו נתון לכם) למחלקות אותן ממשתם (מסומן גם בtodo). הקובץ נתון לנוחיותכם ואתם רשאים לשנות אותו- אך שימו לב שאתם שומרים את הטוקנייזר המאומן בפורמט הנתון, ושהקבצים האחרים (כגון test_tokenizer.py וtrain_ner_model.py מצליחים לקרוא אותו מבלי להכיר את המחלקה שלכם (רק את המחלקה ממנה ירשתם).

בנוסף - עליכם להוסיף למחלקה את האיבר space_token, עם התו שבחרתם עבורו, לדוגמא:

```
class BPETokenizer(BaseTokenizer):  2 usages
    """
    Fixed BPE (Byte Pair Encoding) Tokenizer imp
    """

    def __init__(self, vocab_size: int = 10000):
        """
        Initialize BPE tokenizer

        Args:
            vocab_size: Maximum vocabulary size
        """
        super().__init__()
        self.space_token = "_"
```

שימו לב כי קיים הבדל בין '__' (הנלמד בהרצאה) לבין '_'.

שימו לב! - באפשרותכם להשתמש בדאטה שסופק לכם בלבד עבור האימון. אין להשתמש בנתונים נוספים ממקורות חיצוניים.

עליכם לספק בהגשה ממשק הרצה לאימון ושמירה של שלושת הטוקנייזרים שלכם - generate_tokenizers.py.

עליכם להגיש גם את שלושת הtokenizers המאומנים, בפורמט הנתון לכם, תחת תיקיית trained_tokenizers.

טוקנייזר ראשון ושני:

השתמשו בקבצי האימון שקיבלתם - קבצי המשימה מגיעים מאותו הdomain.

טוקנייזר שלישי - דומיין נסתר:

השתמשו בקבצי האימון מהטוקנייזרים הקודמים. חשבו כיצד להתמודד עם כך שהם נדגמו מdomain שונה מזה שתבחנו עליו.

מבחן (Test):

עבור שני הדומיינים הנתונים דווחו את מהירות ויעילות התיוג על קובץ הdev, ואת ציון הF1 על dev.tagged.

תחרות:

שלושת הטוקנייזרים יבחנו על קבצי תחרות נסתרים כפי שפורט לפני. שימו לב כי לכל רכיב משקל זהה - וייתכן כי קיים טרייד אוף בין הרכיבים הללו.

סביבת עבודה:

לכל זוג הוקצתה מכונה בה מותקנות הספריות הנדרשות לתרגיל. על התרגיל לרוץ בסביבת הקונדה azureml_py38. אין להתקין ספריות נוספות לסביבה זו ללא אישור מסגל הקורס דרך המייל.

הגשה:

קובץ zip בלבד, בשם HW2_123456789_987654321.zip (עבור שני סטודנטים שמספרי הזהות שלהם הם 123456789 ו- 987654321). הקובץ הנ"ל יכלול:

1. **דו"ח קצר** (עד עמוד אחד בפורמט PDF) המכיל הסברים תמציתיים, דיווח וניתוח תוצאות. שם הקובץ צריך להיות report_987654321_123456789.pdf. הדו"ח צריך לכלול:
   a. שמות המחברים ות"ז
   b. אימון - הסבר על כל טוקנייזר שמימשתם. במה הוא שונה מהאלגוריתם הבסיסי, ואיך התמודדתם עם הדומיין הנסתר.
   c. מבחן - דיווח מדד F1 על קובץ ה development עבור כל אחד מהמודלים, ויעילות הטוקנייזרים. כמו כן העריכו מה יהיה ביצועכם על הדומיין הנסתר.
2. **קבצי הקוד של התרגיל**. על הקוד להיות מתועד וקריא. בנוסף, הקוד צריך להיות מסוגל לרוץ על כל מכונה שהיא. אנא כתבו ממשקי הרצה פשוטים לאימון. על הקוד להיות בתיקיית code
3. **ההטוקנייזרים המאומנים**– על הטוקנייזרים להיות בתיקיית trained_tokenizers ולהיות שמורים בשם tokenizer_1.pkl בהתאמה לדומיין (2,3).
4. **ממשק לאימון כל הטוקנייזרים** - על הטוקנייזרים להיות ניתנים לשחזור (Reproducible). שימו לב שאתם יכולים להניח שהdata שניתן לכם הוא זה שנמצא בתיקייה בלבד.

העתקות:

בשל אופי המשימה והמורכבות שלה, קל לבדוק העתקות של קטעי קוד \ קבצים מלאים. למען הסר הספק אנו מדגישים כי אין להעביר קוד בין סטודנטים, בין אם להגשה ובין אם לא.

כפי שמצויין בסילבוס, באפשרותכם להשתמש בכלי בינה מלאכותית (כגון chatGPT) - אך הקוד הוא באחריותכם ועליכם לוודא את תקינותו ונכונותו. כמו כן, גם את קטעי הקוד שנוצרו ע״י שימוש בבינה מלאכותית או את הפרומפט ששימש כדי ליצור אותם אין להעביר בניכם.