שאלות פתוחות

שאלה 1.1

1. **SQuAD**

בהינתן שאלה עבורה נתונה פסקה, הדאטה סט מודד את היכולת של מודל להבין את מבנה ההקשר בטקסט הנתון ולזהות את מיקום התשובה המדויק בפסקה.

2. **HotpotQA**

דורש יכולת קפיצה בין חלקים שונים על מנת לענות על השאלה על ידי בניית חוט מחשבה מקשר.

3. **DROP**

בודק את יכולת ההבנה וההסקה של המודל באמצעות שאלות הדורשות חישובים, ספירה והסקה לוגית מתוך פסקה נתונה, ולכן הוא מודד יכולת הבנה שפתית פנימית.

שאלה 1.2

סעיף א

- **Self-Consistency**
  - **תיאור**

ביצוע מספר CoT ובחר את התשובה לפי דעת הרוב (התשובה שהכי הרבה ריצות CoT הגיעו אליה בדרכים שונות).

  - **יתרונות**
    - משפר את אמינות הפלט (הסתברות נמוכה לתוצאה שגויה זהה על ידי תהליכי חשיבה נפרדים) ואת עמידותו, למשל אם המודל טועה ב20% מהתשובות שלו (שזה הרבה) על ידי הצבעת רוב של מספר CoT ההסתברות שיטעה יותר פעמים משיצדק צונחת.
  - **צוואר בקבוק**

דורש מספר CoT בשביל תשובה יחידה ולכן מגדיל משמעותית את העלות החישובית והזמן הדרוש לתשובה אחת.

  - **מקביליות**

כן ניתן להריץ במקביל מספר CoT.

- **Verifiers**
  - **תיאור**

לאחר שהפלט נוצר, הוא מועבר לכלי שבודק האם הפלט תקין/נכון לפי היגיון ספציפי לסוג המשימה שהמודל נדרש לבצע. ניתן לשלב עם Self-Consistency על ידי בחירת דעת הרוב של הפלטים שאושרו על ידי verifier. ניתן גם להשתמש כחוליה בשרשרת

של CoT (בחירת האפשרות הטובה ביותר מבין אלו שאושרו על ידי verfier ואז להמשיך לג׳נרט את הCoT).

- **יתרונות**
  - מאפשר לבוני המודל לכפות חוקיות מבנית על תשובות לפי רצונם.
  - מודולורי – ניתן לשלב כאמור בתור תהליך ג׳נרוט הפלט.
  - מייצר פלט אמין יותר שעבר את הבדיקות שבוני המודל הגדירו בצורה ישירה כך שיש פחות סיכון שתוחזר שגיאה מפתיעה מהקופסה השחורה שהיא הLLM.
- **צוואר בקבוק**

דורש מעבר לחלק אחר במודל שאינו בהכרח מוטמע ממש בתוך הNN (למשל regex כמו שרועי ציין) מה שיכול להיות מאוד יקר בזמן (העתקת זיכרון ממקום למקום במחשב/שרת).

- **מקביליות**

ה-Verifier עצמו לא יכול לקרות במקביל לפלט של המודל אותו הוא מקבל כקלט ובמובן הזה החישוב מבוצע בטור. עם זאת ניתן להעביר את הפלט במקביל למספר verifiers שונים שבודקים דרישות שונות על הפלט, ואפשר כמובן למקבל את אותו verifier כחוליה בתהליכי CoT שונים שקורים במקביל (כפי שציינתי קודם).

- **Many outputs from many small models instead of one output from a single big model**
  - **תיאור**

במקום להגדיל מודל פי M ניתן להשתמש במודל הקטן M פעמים (במקביל) וכך לייצר M פלטים בעבור אותו כוח חישוב, כאשר הסיכוי שלפחות אחד מהם יהיה טוב עולה.

  - **יתרונות**
    - מייצר יותר פלטים בעבור אותו כוח חישוב, כאשר ההסתברות שחלק מהם יהיו טובים גבוהה, ועל ידי העברת כולם במקביל לverifiers ניתן לקבל פלט אמין.
    - בפועל שיטה זו נותנת תוצאות טובות יותר עבור חלק מהמשימות (למשל חישובים עם נקודה צפה), אך לא תמיד.
  - **צוואר בקבוק**

הפלטים השונים נוצרים בנפרד אך בסוף צריכה להתבצע איזושהי אינטגרציה בינהם כך שעלול להיווצר צוואר בקבוק שם, אף כי כאמור ניתן למקבל את הבדיקה שלהם ופשוט לקחת באופן אקראי את אחד הפלטים שעברו את הverifier ללא אינטגרציה.

  - **מקביליות**

כן, כל עוד אין צורך באינטרגרציה בין הפלטים, אלא יצירה נפרדת שלהם, בדיקה נפרדת שלהם בverifier ובחירה של אחד התקינים.

- **השקעה בזמן ריצה בזמן החיזוי לפי O1 Model ולכאורה לפי R1 Model**
  - **תיאור**

לאפשר למודל לייצר פלט ארוך, או לרוץ זמן רב יותר תוך בחינה עצמית.

  - **יתרונות**
    - מראה שיפור דרמטי בכללי, ואף שיפור רב יותר מאשר אימון נוסף (מרמה מסוימת).

- מאפשר למודל לבדוק את עצמו, לזהות שגיאות, לפרק שאלות מורכבות ולהבין אם הדרך הנוכחית היא שגויה ולהציע רעיונות נוספים וכיוונים חלופיים.
- אדגיש שתחת מתודה זו ישנן מספר תתי מתודות (Back-tracking, self-evaluation, complex CoT) שכולן מתאפשרות בזכות הגישה הכללית של השקעת משאבים בזמן ריצה ארוך יותר, להבדיל מהשקעת משאבים באימון נוסף.

- **צוואר בקבוק**

מעבר למקביליות המוגבלת בה אגע בסעיף הבא, שיטה זו המייצרת פלט גדול, דורשת מכפלות וקטורים ומטריצות הולכות וגדלות כאשר יש פלט ארוך יותר (למשל בשלבי הattention) כך שנדרש מהמודל יכולת לעבד קונטקסט גדול יותר.

- **מקביליות**

בגלל אופן עבודת המודל בו הוא הוא יוצר טוקן אחד בכל פעם והצורך בשיטה זו לבצע בדיקה למה שהוא כתב עד כה, לא ניתן למקבל אותה. כלומר – המודל לא יכול לבדוק מה הוא כתב עוד לפני שהוא כתב זאת, ואם הוא יודע שהדרך שהוא הולך בה תגיעה למבוי סתום אז מלכתחילה הוא לא ילך בה וזה לא החידוש שמביאה שיטה זו.

סעיף ב

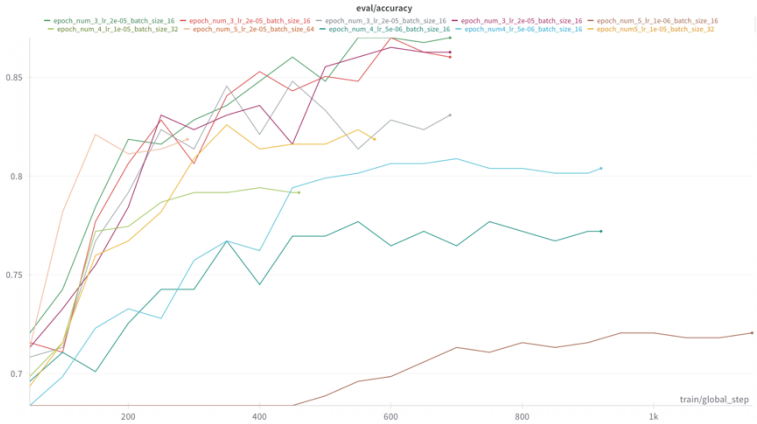
הייתי בוחר בSelf-Consistency, וכתלות בסוג השאלה הייתי משלב verifiers.
בשיטה זו ניתן למקבל חישובי CoT רבים ולנצל באופן מייטבי את הGPU הגדול שיש לראשותנו. וכמרכיב מקובל בשרשרת (כפי שציינתי) הייתי משלב verifiers בתוך התהליך אם אופי השאלה מתאים. מאחר וזו ***שאלה מדעית*** הכוונה ב-"אופי השאלה" היא מצד אחד לאכוף חוקי טבע מדעיים באופן נוקשה בחלקים בהם נדרש דיוק לא מתפשר (למשל שלא יעגל את π ל3 כי הוא ראה את זה בתרגילים באיזו חוברת), ומצד שני לאפשר למודל יצרתיות בחלקים בהם הוא נדרש לתת הסבר חדשני לבעיה מדעית ולא להגביל אותו יותר מדי (למשל אם היינו רוצים מודל שימציא את הרעיון של מספרים מרוכבים אם לא היה קיים, היינו צריכים לא לאכוף באופן נוקשה מדי את האיסור על שורש של מספר שלילי).

חלק תכנותי

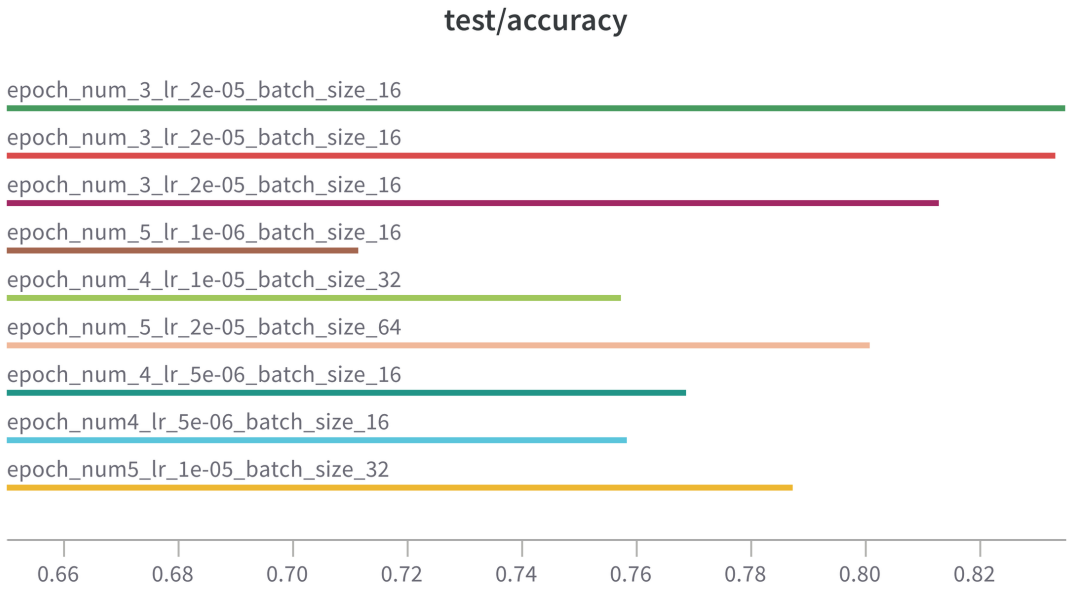
קישור לפרויקט ב**GitHub**

https://github.com/Yam-Arieli/mrpc-finetune-exercise

האם הקונפיגורציה שהשיגו את הדיוק הטוב ביותר בואלידציה השיגה גם את הדיוק הטוב ביותר במבחן?



כן, כפי שאפשר לראות שלושת הואלידציות העליונות (אשר כולן עם אותה קונפיגורציה) הן גם הגבוהות גם במבחן, ובפרט זו בצבע ירוק כהה. הסיבה שהרצתי 3 פעמים היא שראיתי בהרצות גם את הגרף האפור שהינו עם אותה קונפיגורציה אך הראה ביצועי accuracy נמוכים יותר, אך כשיצרתי אותו לא יצרתי תוצאות מבחן לצערי. ניסיתי לשחזר ביצוע פחות טוב עם קונפיגורציה זו פעם נוספת אך ללא הצלחה. השארתי את תוצאות ריצה זו בכל מקרה בשביל להמנע מcherry picking.



ניתוח איכותני

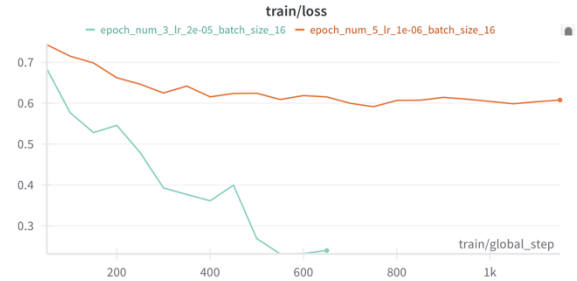
- *השווה בין הקונפיגורציה הטובה ביותר לגרועה ביותר.*
- *השווה דוגמאות ואלידציה בהן הטוב הצליח והגרוע נכשל.*

- *האם ניתן לאפיין דוגמאות אלו?*

```
for ds in [train_dataset, eval_dataset, test_dataset]:
    print(pd.DataFrame(ds).label.mean())
✓ 0.2s

0.6744820065430752
0.6838235294117647
0.664927536231884
```

נראה כי כל דוגמאות הואלידציה בהן המודל הטוב הצליח והגרוע נכשל הן דוגמאות שהסיווג הנכון הוא 0 והמודל הגרוע סיווג 1. אציין כי **הסיווג הנכון עבור כשני שליש מהדוגמאות בכל אחד מהסטים הוא 1**, כפי שניתן לראות בתמונה המצורפת. סביר שהמודל הגרוע נפגע באופן חמור יותר מהטיה זו – הוא בעל קצב למידה קטן יותר והוקצאו לו יותר epochs. ככל הנראה המודל הגרוע בחר בכיוון זה כ"פשרה סטטיסטית" שהוא מבצע עקב חוסר יכולת לדלג לאזור טוב יותר בפרמטרים, כלומר underfitting ולא עקב overfiting כפי שמשתקף בגרף המצורף בו שגיאת האימון מפסיקה לרדת בשלב מוקדם יחסית ונשארת גבוהה (המודל לא מצליח ללמוד משהו חדש).



נראה כי דוגמאות מסוג זה מאופיינות במבנה דומה של המשפט, ואף רישא זהה. הסיבה לחוסר השיוויון בין המשפטים נובע מ:

- מידע נוסף שקיים במשפט אחד ולא בשני כך של ניתן להגיד שהם אומרים את אותו הדבר, אף על פי שהדבר נכון בחלק מהמשפט.
- משפטים שהגיוני שילכו ביחד (באחד הS&P 500 יורד ובשני הNASDAQ למשל) ולמרות שהקורלציה בינהם הגיונית, הם אינם בעלי אותה משמעות.
- הבדל מספרי שנכתב במילים (three במקום 3).