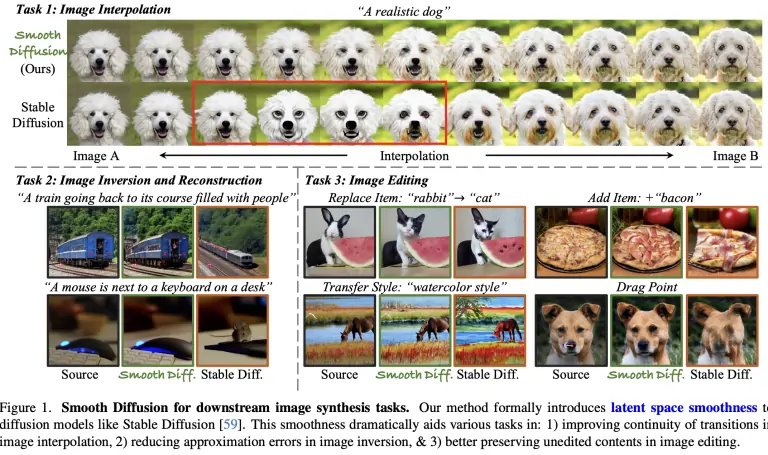
## **Review 181, Short: Smooth Diffusion: Crafting Smooth Latent Spaces in Diffusion Models**

<https://huggingface.co/papers/2312.04410>

****

בסקירתנו היום נדבר איך אנחנו יכולים ״לסדר״ את המרחב הלטנטי של מודלי דיפוזיה גנרטיביים. המאמר מנסה ״לסדר״ את המרחב הלטנטי של מודלי דיפוזיה. בשביל להבין מה זה המרחב הלטנטי של מודלי דיפוזיה הוא למעשה מרחב של וקטורים גאוסיים סטנדרטים שהמימד שלהם שווה למימד שאנקודר מקודד כל תמונה אליו. למי שכח על מה אני מדבר כאן אזכיר כי מודלי דיפוזיה לטנטיים מייצרים ייצוג לטנטי של תמונה על ידי ניקוי הדרגתי (באיטרציות) של הרעש מוקטור גאוסי סטנדרטי (backward process).

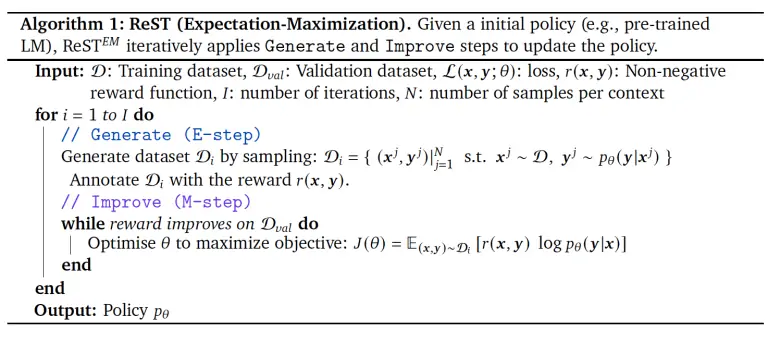
לאחר השלמת התהליך מעבירים את הווקטור שנוצר דרך רשת הדקודר ליצירת תמונה. המאמר מנסה לגרום לכך ששינוי קטן בוקטור הלטנטי שנוצר על ידי מודל דיפוזיה יוביל לשינוי קטן בתמונה הנוצרת. זה חשוב כי זה נותן לנו אפשרות לשלוט בצורה יותר טובה במה אנחנו מייצרים עם המודל וגם מאפשר לנו ליצור ״מעברים חלקים״ בין התמונות השונות. אז מה הם עשו?

בגדול הרעיון שהזזה של הוקטור הלטנטי למרחק d תרגום להזזה שהיא לכל היותר cd בתמונה שנוצרת ממנו כאשר c הוא קבוע (לא תלוי בתמונה). מכיוון שקשה לכפות את זה באופן ישיר במהלך אימון המודל המאמר בחר להשתמש בטכניקה נפוצה של רגולריזציה מעולם הגאנים(GANs). ניתן להראות כי פונקציית לוס האוכפת יעקוביאן (מטריצה נגזרות) ביחס לוקטור לטנטי מוכפל בשינוי בתמונה הנוצרת(J\*dx) להיות קבוע משיגה את המטרה המיוחלת.

למעשה זה קירוב טיילור מסדר ראשון של התמונה הנוצרת על ידי הזזה של וקטור לטנטי. מעשית מוסיפים איבר לפונקציית לוס הרגילה של מודל דיפוזיה שקונס על אי התאמה J\*dx לממוצע המעריכי שלו על פני האיטרציות הקודמות של gradient descent. זה נשמע קצת לא פשוט אבל הנוסחאות במאמר לא מורכבות יותר מדי…

## **Review 182: Beyond Human Data: Scaling Self-Training for Problem-Solving with Language Models**

<https://arxiv.org/abs/2312.06585>

****

היום אנו מדברים על אימון של מודלי שפה. באחת הסקירות האחרונות הסברתי לכם איך מאמנים מודל שפה עם RLHF (למידה עם חיזוקים המשולבת עם משוב אנושי) ולמה זה נחוץ. אזכיר לכם כי צריך להשתמש בטכניקות של RL כדי לאמן מודל בתנועה כלומר לעדכן משקלים של המודל על הדאטה שנוצר אחרי העדכון האחרון של המודל. במונחי RL הדאטה נדגם לפי ה-policy הכי מעודכן (כלומר למידה on-policy).

אימון מודלי שפה עם RLHF בצורת on-policy הוא יקר (כל הזמן צריך ליצור דאטה) ולא תמיד יציב ולכן הוצעו מספר שיטות חלופיות פחות כבדות כמו DPO ו-REST. הרעיון ב-REST הוא לא ליצור דאטה חדש בצורה on-policy אלא:

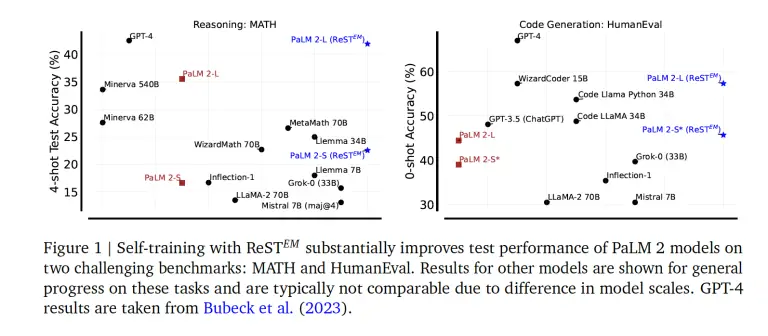
* ליצור דאטהסט באמצעות מודל התחלתי ולשלב אותו עם דאטה המתויג על ידי בני אדם
* לבחור את הדאטה בעלת ערכי פונקציית תגמול גבוהים מעל סף התחלתי
* לאמן את המודל עם הדאטה הזה
* ליצור עוד דאטה עם המודל המעודכן (כל I איטרציות אחרת לדלג על השלב)
* ליצור(לסנן) דאטה עם ערך(תגמול) גדול מערך סף גבוה יותר
* לאמן מודל עם דאטה חדש…

המחברים לקחו את הרעיון הזה ושכללו אותו (מבחינת הביצועים) בהתבסס על הרעיון של (Expectation-Maximization(EM ולייתר שימוש בדאטה הנוצר על ידי בני אדם. הרעיון ב-EM הוא למקסם נראות מירבית של פונקציית הסתברות p ביחס לפרמטרים כאשר הדאטה נדגם מהתפלגות אחרת q. זה מורכב משני שלבים איטרטיביים:

* E: מקרבים את q מבחינת KL Divergence (אופטימיזציה)ֿ
* M: ממקסים את נראות מירבית (עם דאטה הנדגמת עם q) ביחס לפרמטרים.

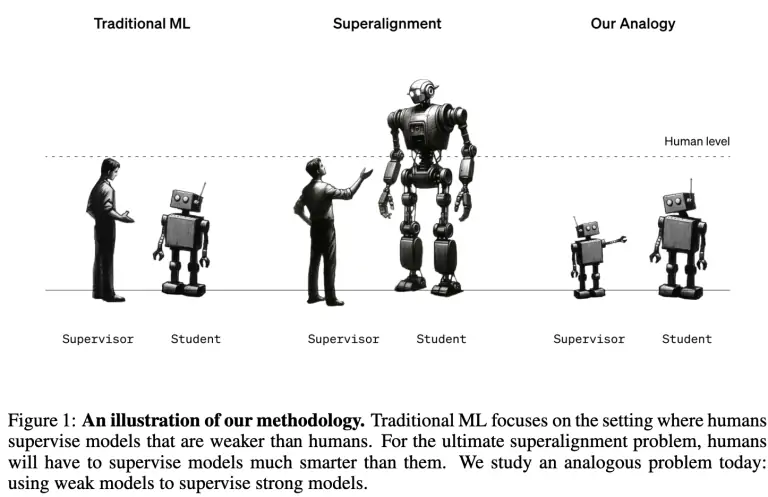
אז המחברים לקחו את הרעיון הזה והפעילו אותו על RL בצורה הבא:

1. ליצור דאטה מהמודל
2. עד שהתגמול בסט ולידציה עולה:
   * לאמן מודל בסיס (תמיד מאמנים מודל בסיס להבדיל מ-REST) כאשר כל דגימה בדאטהסט ממושקלת עם ערך התגמול. מכיוון שפונקציית התגמול במאמר היא בינארית זה שקול לאי התחשבות בדוגמאות בעלי ערך פונקציית תגמול 0
3. חוזרים לשלב 1 מספר איטרציות



## **Review 183,** **Short: WEAK-TO-STRONG GENERALIZATION: ELICITING STRONG CAPABILITIES WITH WEAK SUPERVISION**

<https://openai.com/research/weak-to-strong-generalization>, <https://cdn.openai.com/papers/weak-to-strong-generalization.pdf>

****

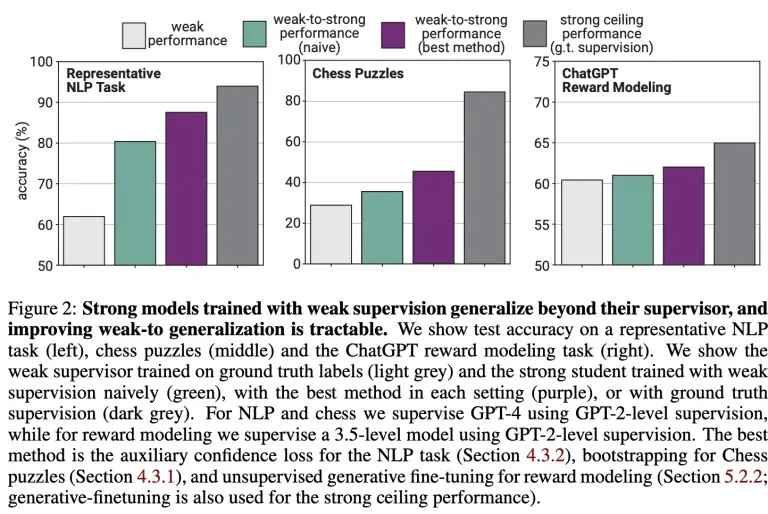
אוקיי, הסקירה של היום על מחקר מאוד מעניין מבית לא אחר אלא OpenAI. המאמר מנסה לפתח שיטות אימון (כיול) למודלים חזקים עם מודל פיקוח חלש. מה זה בעצם אומר? נניח שיש לנו מודל S בעל יכולות חזקות יותר (נגיד במספר פרמטרים) ממודל חלש יותר W ובנוסף יש לנו דאטהסט לא מתויג שאנחנו רוצים לכייל על מודל S.

פיקוח חלש אומר שאנו קודם כל מתייגים את הדאטה עם W ואז מאמנים מודל חזק S עם הדאטהסט המתויג הזה. למה זה בעצם חשוב? המאמר מדבר בעתיד הקרוב יחסית אנו נגיע לאימון של מודלים בעלי יכולות superhuman למשימות שאנו בני אדם לא מסוגלים לבצע באיכות טובה ואז אנחנו בעצם מהווים את המודל החלש W.

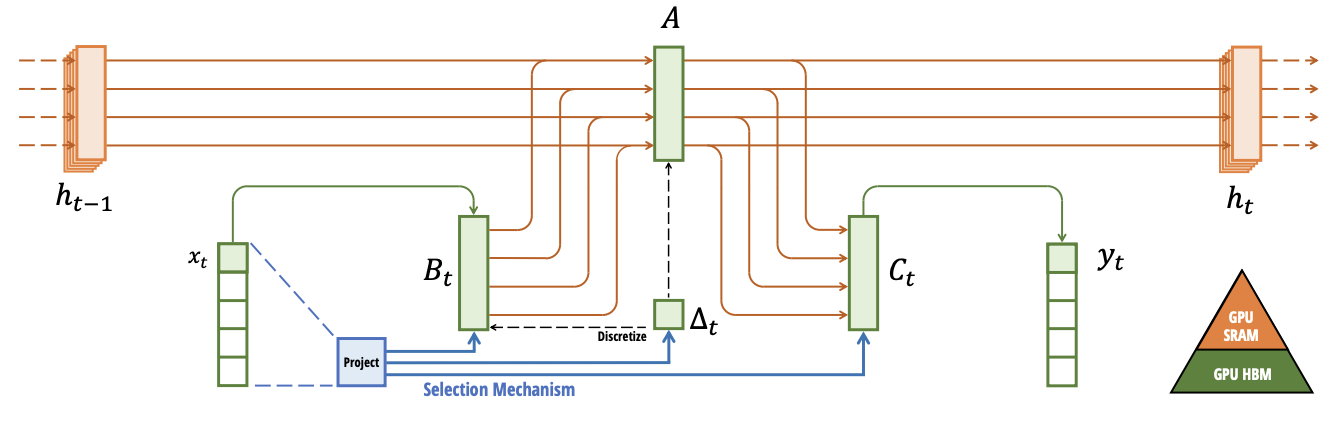
השאלה עד כמה אימון מודל בפיקוח חלש עובד גרוע יחסית לכיול של מודל S עם דאטה מתויג נכון (על ידי בני אדם)? מתברר שההפרש בביצועים הוא די גדול למרות שאימון בפיקוח חלש כן מצליח לשפר את הביצועים של המודל החזק.

השאלה בעצם האם יש שיטות שמשיגות ביצועים טובים יותר מאשר אימון בפיקוח חלש? המאמר מציע שתי שתי משפחות של שיטות. השיטה הראשונה היא אימון הדרגתי של המודל החזק. מתחילים ממודל חלש וכל פעם ״מחזקים״ אות בקצת (בכמות הפרמטרים למשל) כאשר המודל מהאיטרציה הקודמת משרת בתור מודל פיקוח חלש. הגישה השנייה (consistency loss)היא לתת פחות לקנוס את המודל החזק על אי התאמה עם המודל החלש כאשר המודל החזק מאוד בטוח בתוצאה שלו.

יש תוצאות מעניינות…

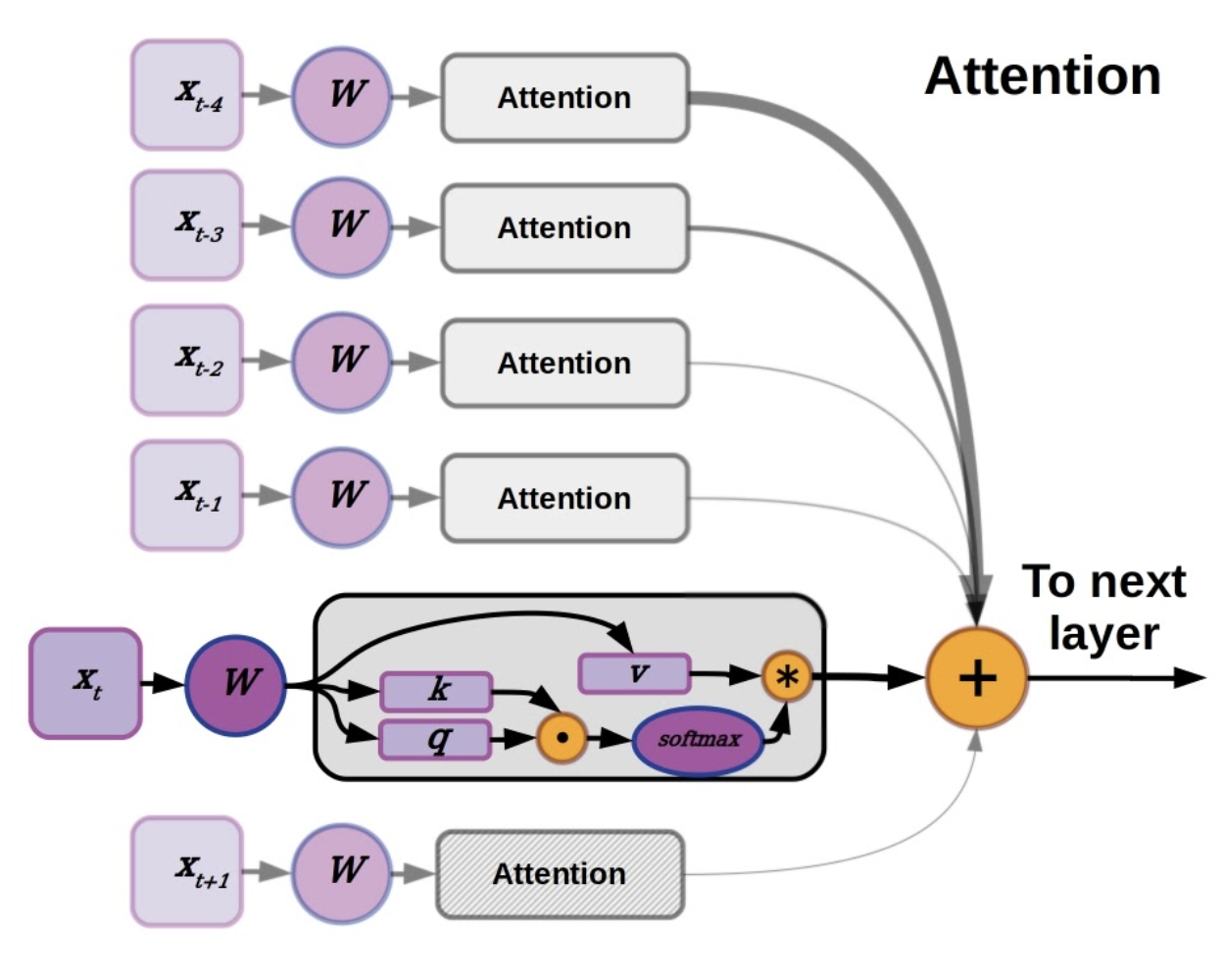


## **Review 184: Mamba Series, An Intro**

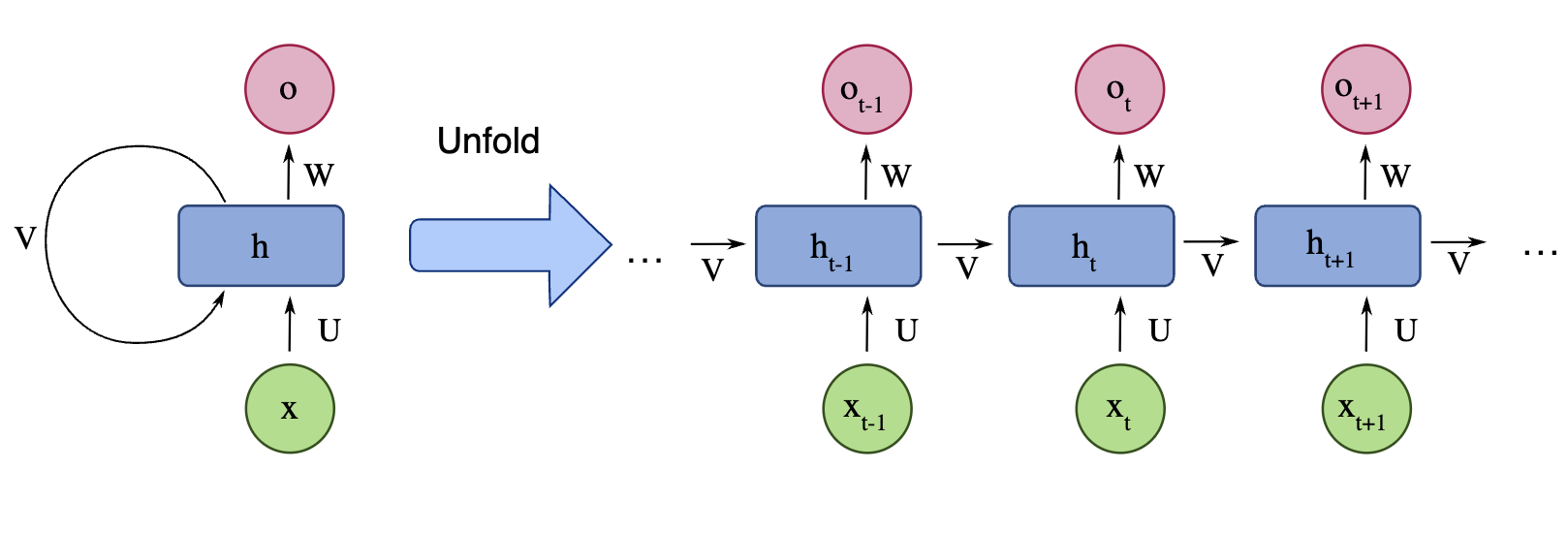


כמו שהבטחתי היום אנחנו מתחילים את סדרת הסקירות הקצרות שתוביל אותנו בסופו של דבר לדובדבן שבקצפת שזה תיאור של ארכיטקטורה בשם ממבה (Mamba) שעשתה הרבה רעש לאחרונה. היום אספק לכם מבוא כללי על מה שאנחנו הולכים לדבר בשבוע הקרוב.

אז מה זה ממבה? כאמור ממבה הינה ארכיטקטורת רשת נוירונים שבאה לתת מענה לחיסרון הבולט של הטרנספורמרים והוא הסיבוכיות הריבועית במונחי אורך הקלט (עבור קלט טקסטואלי אורכו הינו מספר הטוקנים בחלון ההקשר). נכון שלאחרונה הוצעו מספר שיפורים למנגנון תשומת הלב של הטרנספורמר (שהוא לב הבעיה) כמו FlashAttention2 אבל עדיין הטרנספורמרים מתקשים לעבוד בצורה המיטבית עם דאטה בעל אורכי הקשר בסדר גדול של מיליוני טוקנים.



אז איך אנחנו נתמודד עם הסיבוכיות הריבועים של מנגנון תשומת הלב של הטרנספורמרים? הצעירים שבינינו זוכרים שפעם היתה לנו ארכיטקטורה הנקראת (RNN (Recurrent Neural Networks שבהם לא היתה לנו בעיה של הסיבוכיות הריבועית (היו שם חסרונות אחרים שנדון בהם בהמשך). ה-RNN והשכלולים שלו כגון GRU ו-LSTM לא היו צריכים לקחת בחשבון את ייצוג של כל פיסות דאטה (נגיד טוקנים בטקסט) בצורה מפורשת (כפי שמנגנון תשומת הלב של הטרנספורמים עושה). במקום זאת הוא היה דוחס את המידע על הטוקנים (הייצוגים והקשרים ביניהם) באמצעות וקטור המצב (ב-LSTM יש בנוסף עוד כמה וקטורים האחראים על דחיסה של זכרון).



אז אם הצלחנו לדחוס את כל המידע הטמון בטוקנים הקודמים **בצורה טובה** אז לא צריך להתחשב שום מידע באופן מפורש בזמן אימון ובזמן היסק (inference) של טוקן הבא. אמנם אם כל הזכרון שלנו נמצא בוקטור הדחוס הזה אז נשתמש בו במקום להתחשב כל הטוקנים הקודמים. אולם יש בגישה הזו שתי בעיות עיקריות:

1. ארכיטקטורות RNN **לא הצליחו לדחוס בצורה טובה את הטוקנים הקודמים** כאשר אורך חלון הקשר ארוך וזו הייתה הסיבה העיקרית שארכיטקטורות אלו נכשלו במשימות הכרוכות בעיבוד קטעי טקסט ארוכים. הרי אם המודל לא מסוגל לזכור את המידע מהטוקנים הקודמים, לא ניתן לצפות ממנו ביצועים טובים בחיזוי טוקן הבא.
2. בטח כבר שמעתם שארכיטקטורות RNN **הן לא scalable**. מה זה בעצם אומר? כאשר אנו מבצעים אימון של מודל שפה המשימה היא לחזות חלקי קלט שאנו מסתירים (ממסכים) מהמודל. עם טרנספורמרים יש לנו יכולת לחזות את כל הטוקנים הנסתרים בצורה מקבילית עלי ידי שימוש בו זמני במסיכות שונות (כל פעם ממסכים רק את מה שצריך). ב-RNNs זה בלתי אפשרי כי עבור חיזוי של כל טוקן אנו צריכים לחשב את ייצוג הזכרון שלוקח בחשבון את כל הטוקנים שהיו לפניו. כלומר עבור טוקן מספר 1000 אנו צריכים לבצע חישוב סקוונציאלי (אחד אחרי השני) עבור 999 טוקנים שהיו לפניו. פעולה זו לא ניתנת למקבול עקב נוכחות של פונקציות לא לינאריות בחישוב ייצוג הזכרון (ממבה עוקף את המכשול הזה באלגנטיות). כמובן שזה לא יעיל ומהווה מכשול משמעותי בניצול יעיל של משאבי חישוב (GPUs).

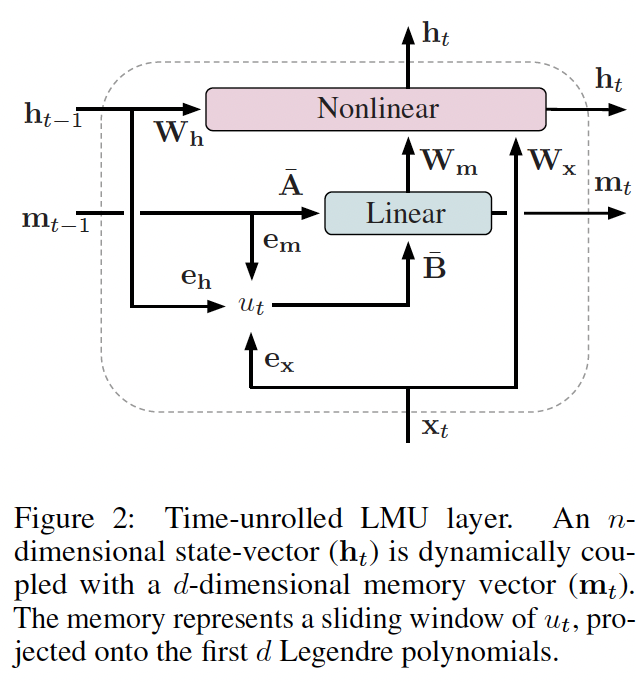
עכשיו נשאלת השאלה האם אנחנו יכולים להתגבר על הסיבוכיות הריבועית של הטרנספורמרים ובאותו זמן להיפטר משני חסרונות שתיארנו בפסקה הקודמת? זו בדיוק השאלה המחקרית העיקרית שנדון בה בסקירות הבאות שיובילו אותנו לארכיטקטורת ממבה הנחשקת.

לבסוף אתן לכם כמה טיזרים קטנים בנוגע למה שאתם הולכים לראות בימים הקרובים:

1. ארכיטקטורות שאנו הולכים לדבר עליהן מאפשרים שני משטרי הפעלה:
   1. אימון מקבילי כמו עם טרנספורמרים במהלך אימון המודל
   2. היסק מהיר כמו עם ה-RNN-ים
2. באופן די מפתיע ארכיטקטורה זו היא בעלת מבנה דומה לרשת קונבולוציה רק שהקרנלים של קונבולוציות אלו הן מאוד ארוכות
3. ארכיטקטורות אלו שואבות השראה ממערכות דינמיות לינאריות וקשרות לשערוך של פונקציות על ידי פולינומים אורתוגונליים.

## **Review 185: Legendre Memory Units: Continuous-Time Representation in Recurrent Neural Networks**

https://proceedings.neurips.cc/paper\_files/paper/2019/file/952285b9b7e7a1be5aa7849f32ffff05-Paper.pdf



המאמר הראשון בסדרה שלנו מנסה לטפל בעיה הראשונה של RNNs כלומר באי יכולת של רשתות אלו לדחוס את הזיכרון (קלט בחלון ההקשר) בצורה מספיק טובה. המאמר מציע גישה מקורית ומעניינת שמקורה במערכות דינמיות (Dynamic Systems) לבניית ייצוג הזיכרון. נניח שיש לנו פונקציית קלט רציפה (u(t ואנו רוצים לבנות מערכת ש״זוכרת את הפונקציה זו״ כלומר בונה ייצוג כך שיהיה אפשר לשחזרה באופן מדויק. תזכרו שכדי לתאר קלט דיסקרטי כמו טקסט אנו צריכים רק לעשות דיסקרטיזציה או לדגום את הפונקציה הזו.

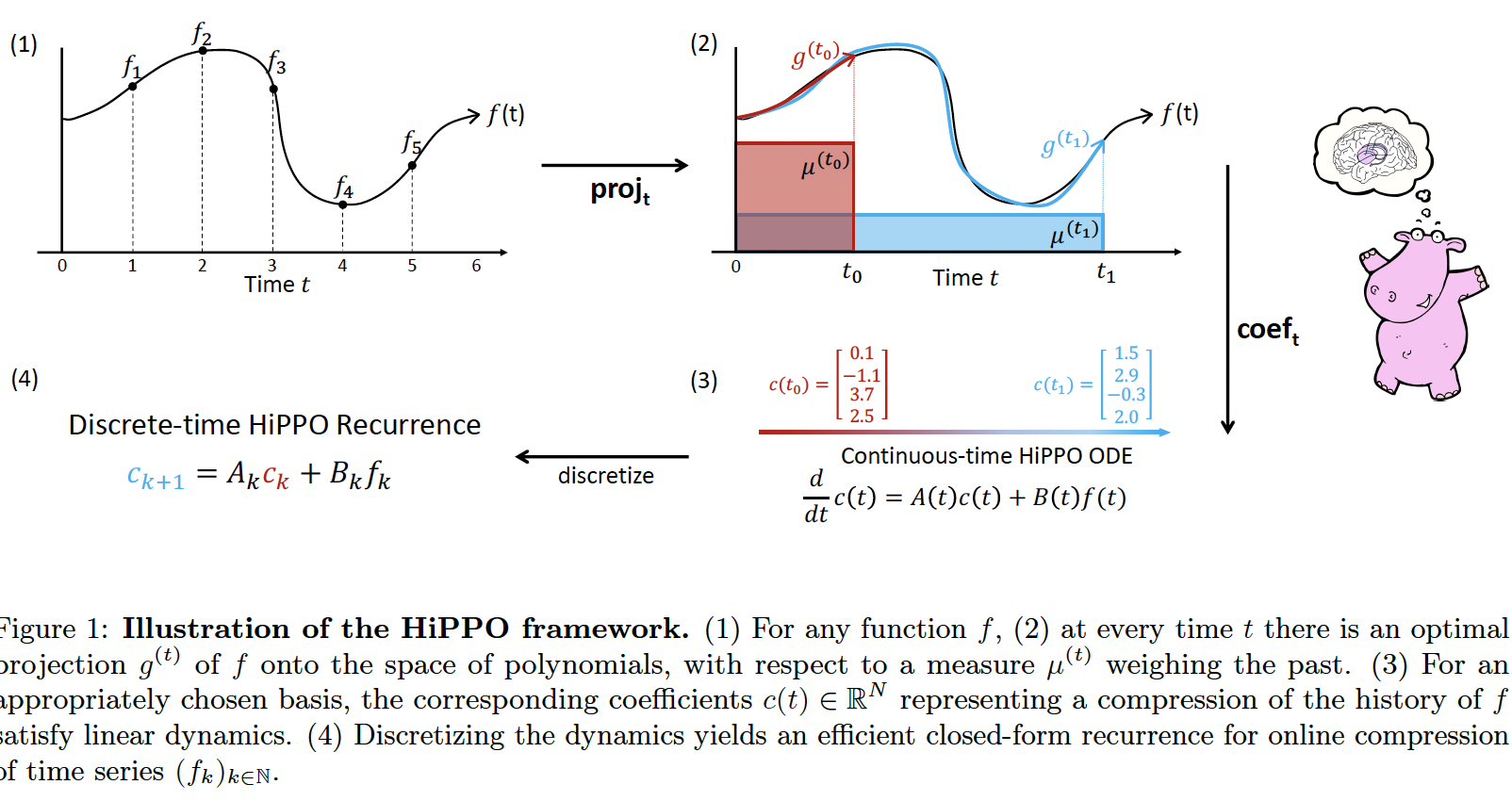
המאמר בונה מערכת דינמית המתוארת על ידי משוואה דיפרנציאלית לינארית (מערכת דינמית, משוואה 1 במאמר) כאשר (m(t הוא וקטור הזכרון ו- (u(t כאמור הקלט (כרגע חד ממדי). מתברר שעבור בחירה מסוימת של מטריצת A במשוואה של המערכת הדינמית ניתן לתאר את הקלט (בפרק זמן מסוים) על ידי שילוב של פונקציית הזיכרון (m(t ופונקציות מתמטיות הנקראות פולינומים של Legendre (משוואה 3 במאמר). כלומר ניתן לתאר את כל מה שקרה מבחינת הקלט עד זמן מסוים על ידי פונקציה (m(t - וזה בדיוק מה שרצינו, נכון?

אולם הדאטה שלנו דיסקרטי (טוקנים נגיד) אז צריך לעשות דיסקרטיזציה (דגימה) לגישה הזו. כלומר במקום פונקציות רציפות תהיה לנו סדרת הקלט u\_t וקטור הזיכרון m\_t. גם מטריצות במערכת הדינמית שלנו צריכות לעבור דיסקרטיזציה (השערוך הרגיל של הנגזרת/גרדיאנט) ואז נקבל נוסחה רקורסיבית עבור m\_t כפונקציה של u\_t ו- m\_t-1. ניתן לתאר את את הדגימות עד t=T על ידי נוסחת נסיגה הזו.

זהו זה - יש לנו רשת בסגנון RNN כאשר הזיכרון ממודל על ידי דיסקרטיזציה של מערכת דינמית, המחשבת מקדמים של פולינומי Legendre ובאופן זה עבד לא רע אי שם ב 2020.

## **Review 186: HiPPO: Recurrent Memory with Optimal Polynomial Projections**

https://arxiv.org/abs/2008.07669



הגענו למאמר השני בסדרה - המאמר הזה חשוב מאוד כי הוא מפתח בסיס מתמטי מוצק המשמש כל המודלים מבוססים על מערכות דינמיות לינאריות כולל כמובן ממבה. המאמר הזה קצת (די הרבה) כבד מתמטית אך אנסה לעשות כמיטב יכולתי כדי להעביר לכם את המסר העיקרי שהוא מביא איתו.

בסקירה הקודמת דיברנו על איך ניתן לבנות וקטור זיכרון (m(t בעל יכולת לשחזר פונקצית קלט (u(x ל-x מאינטרוול ; כאן t מסמן גודל חלון הקשר (כלומר אורך הזיכרון). פונקצית (m(t ממודלת על ידי מערכת דינמית לינארית ושילובה עם פולינומי Legendre משחזר לנו את הקלט u. נעיר שאנו עובדים עם הגרסאות הדיסקרטיות של המודלים האלו שהן בעצם נוסחת נסיגה עבור סדרת וקטורי הזיכרון m\_t.

המאמר המסוקר מנסח מסגרת מתמטית כללית עבור בעיית ייצוג הזיכרון של פונקצית קלט (u(x בתחום . והנה מתחיל הסיבוך: קודם כל פולינומי Legendre הם מקרה פרטי של פונקציות אורתוגונליות במרחב הילברט (יותר נכון מרחב פונקציונלי L של לבג - המקרה הפרטי של הילברט) המצויד בנוסף בפונקציית מידה mu. אוקיי, מה הדבר הזה אומר בעצם? ממש בגדול זה מרחב של פונקציות שהמכפלה הפנימית ביניהן מוגדרת בתור אינטגרל של מכפלתן תחת מידה mu (במקרה הפשוט ביותר מידה mu שווה ל 1 זהותית ואנו מקבלים אינטגרל Riemann רגיל של המכפלה אבל עבור mu מורכבים יותר כמו Riemann-Stieltjes). פונקציות אורתוגונלית במרחב החמוד הזה מוגדרות בתור אלו שהמכפלה הפנימית שלהן שווה ל 0 (תחת מידה mu). פולינומי Legenge הן אורתוגונליים תחת מידה mu השווה ל- ב- ואפס בכל מקום אחר.

אז נניח שיש לנו N פונקציות אורתוגונליות במרחבנו החמוד. ועכשיו המטרה היא לתאר את הקלט (u(x ב- על ידי . כלומר אנו רוצים לבנות סכום ממושקל (u\*(x של עם מקדמים מסוימים (שימו לב שעבור t-ים שונים מקבלים וקטורי מקדמים שונים וכך שיש לנו כאן פונקציה וקטורית של המקדמים התלויה ב-t).

כלומר (u\*(x צריך לקרב בצורה טובה את הקלט (u(x (כלומר למזער שגיאה ביניהן ב-). והדיוק מחושב בתור אינטגרל של ההפרש הריבועי בין (u\*(x ו- (u(x תחת מידה mu (כאמור היא שווה ל- ב- עבור כל x ואפס בכל מקום אחר עבור פולינומי Legendre אבל כמובן קיימות עוד אפשרויות). איך נחשב מקדמים הממזערים את ההפרש הזה? לא כזה מסובך: מקדם i שווה למכפלה פנימית (=אינטגרל) בין פונקציה מספר i לפונקצית קלט u תחת אותה מידה mu.

עכשיו איך כל זה קשור למערכות דינמיות לינאריות החמודות שלנו? מתברר כי מערכת דינמית לינארית שתיארנו בסקירה הקודמת עבור וקטור (m(t מתארת את המקדמים של ייצוג הקלט באמצעות N פולינומי Legendre אורתוגונליים תחת מידה mu שהגדרנו לפני. ו-N זה המימד של וקטור הזיכרון (m(t תחת מידה mu הדורשת קרבה אחידה (=זכרון אחיד) בין u\* ו- u ב- .

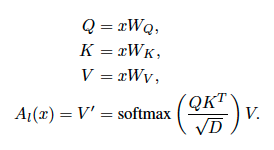
אם נגדיר מידה mu להיות פונקציה (exp(x-t עבור t נתון, מערכת דינמית לינארית אחרת תתאר לנו מקדמים של פולינומי Laguerre (אורתוגונליים תחת mu הזו). שימו לב שמידה זו מגדירה זיכרון הדועך מעריכית כלומר ככל שעובר הזמן מזמן הנוכחי t, הזיכרון הולך ונהיה מעומעם יותר.

בנוסף המאמר מדבר גם על שיטות דיסקרטיזציה של מערכת דינמית זו וגם דן בקשר בינה לבין RNNs.

אוקיי, עכשיו סיכום במשפט אחד של המאמר הדי כבד הזה. המחברים בנו מסגרת מתמטית למידול בעיית הזיכרון של פונקציית קלט שישמש אותנו מאחורי הקלעים לבניית מודלי attention כל הדרך לממבה.

## **Review 187: Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention**

https://arxiv.org/pdf/2006.16236



אחרי הסקירה הקודמת הכבדה מאוד מחכה לנו היום סקירה קלילה (הסקירה הבאה הולכת להיות די כבדה). כמו שכבר אמרנו אחד החסרונות הבולטים של הטרנספורמר היא הסיבוכיות הריבועית שלו במונחי אורך הקלט (= מספר איברים בסדרת הקלט). הסיבוכיות הזו בא על ידי ביטוי גם במהלך האימון וגם במהלך ההיסק (inference). סיבוכיות ריבועית זאת כואבת במיוחד בזמן ההיסק כאשר אין לנו יכולת לחזות מספר טוקנים בו זמנית כי לחיזוי טוקן n אנו צריכים לדעת את ה-(n-1) הטוקנים הראשונים. האם ניתן להפוך את הטרנספורמר לסוג של RNN במהלך ההיסק כאשר כל הזיכרון על הטוקנים הקודמים נדחס לכמה וקטורים בודדים (וקטור זכרון ווקטור של המצב)?

הטרנספורמר המקורי אינו מאפשר אופן חישוב כזה כי הוא מכיל פעולה לא לינארית (softmax) בתוך מנגנון תשומת הלב שלו. ניתן לראות די בקלות שלא ניתן לעקוף את מגבלת הסיבוכיות הריבועית שלו ללא שינוי של אופן חישוב של תשומת הלב. המאמר המסוקר מציע להחליף את חישוב הסופטמקס במנגנון זה בחישוב לינארי (מכפלת מטריצות) המחושבות על ידי הפעלת פונקציה לא לינארית phi על וקטורי השאילתות Q ושל וקטורי המפתחות K. מי שעוד זוכר מה זה KT)Kernel Trick) מבין מה שנעשה כאן הוא KT בכיוון ההפוך.

כמובן שאנו מאבדים כאן מהעוצמה של המנגנון תשומת הלב הרגיל אבל זה יעזור לנו לפתור את סוגיית הסיבוכיות הריבועית בזמן ההיסק. למעשה המחברים מוכיחים (ראו את התמונה למעלה) כי ניתן לממש את המנגנון הזה לסדרתי בעל סיבוכיות לינארית במונחי אורך הקלט. כמובן בזמן האימון ניתן לחשב חיזוי של כמה טוקנים בו זמנית (לפי היכולת החישובית שעומדת לרשותנו) וליהנות מהיתרון של מנגנון תשומת הלב הרגיל.

כלומר יש לנו טרנספורמר (מוחלש כמובן) באימון ו- RNN בהיסק. בהמשך נראה ניתן לשפר את הגישה הזו עם SSMs) state-space models).

## **Review 188: Efficiently Modeling Long Sequences with Structured State Spaces**

https://arxiv.org/abs/2111.00396

לאט לאט הגענו למאמר הרביעי בסדרת סקירות בדרך לממבה. הפעם נסקור מאמר מ-2022 שיצא שנתיים אחרי 3 המאמרים הראשוניים שסקרנו בנושא המעניין הזה. כמובן במהלך תקופה זו יצאו כמה מאמרים מעניינים שפיתחו ארכיטקטורות מבוססות מערכות דינמיות לינאריות (ובשם כללי יותר Space-State Models- SSMs).

המאמר שנסקור לקח את הגישה הזו לגבהים חדשים והגיע לתוצאות די מרשימות עם דאטה בעל אורך הקשר ארוך (למשל עבור אות אודיו המכיל אלפי או אפילו עשרות אלפי דגימות בשנייה. אם יש לנו מטלה שדורשת התחשבות בכמה עשרות שניות של אודיו אז אנו צריכים אורך הקשר של מאות רבות של דגימות וזה די כבד עבור הטרנספורמר עם הסיבוכיות הריבועית שלו - במונחי אורך הקשר).

אוקיי, אז בואו ניזכר מהו היתרון הבולט של ארכיטקטורות מבוססות SSMs. מצד אחד בעת ההיסק (inference) של טוקן הם מונעים מאיתנו צורך להתחשב באופן מפורש בכל הדגימות הקודמות על ידי דחיסה של המידע בטוקנים הקודמים(=זיכרון) בווקטור זיכרון אחד, המתעדכן עם המערכת הדינמית הלינארית. מצד שני במהלך האימון (כשכל הטוקנים ידועים) הוא מאפשר חישוב בו זמני של כל הטוקנים הממוסכים.

דואליות עוצמתית זו התאפשרה על ידי ייצוגה של זיכרון בתור מערכת לינארית שניתן לבטא את הזיכרון המצטבר לכל טוקן כפעולה לינארית. כלומר ניתן לתאר את הפלט של עבור טוקן k על ידי הנוסחה באחת התמונות (הקטנה יותר).

מטריצות בנוסחה הן הגרסאות המודסקרטות של המטריצות המופיעות בנוסחה של המערכת הדינמית המתארת את התקדמות הזכרון בזמן (טוקנים). ניתן לראות כי מה שיש לנו כאן זו רשת קונבולוציה(שעלולה להיות מאוד ארוכה) שמאפשרת חישוב הייצוג של כל טוקן i.

קיבלנו את הארכיטקטורה הדואלית המתאימה גם לאימון וגם להיסק. אבל יש בעיה קטנה. עבור אורך הקשר גדול מספיק נדרשת כמות גדולה מאוד של זכרון. קודם כל אנו צריכים מטריצה A בגודל NxN (נגיד עבור N=64) עבור כל מימד של ייצוג הקלט (כי זה מה שהמערכת הדינמית שלנו ״צריכה לזכור״). אז חישוב קונבולוציה זו בצורה הישירה עבור מטריצה A כללית של המאמר של HiPPO (עבור מקרה של פולינומי Legendre שנקרא LegT תחת מכסה המנוע של המערכת הדינמית) הוא מאוד כבד ודורש הרבה זיכרון.

אז מה ניתן לעשות? קודם כל אם מטריצה A היא אלכסונית החישוב ודרישות הזכרון היו הרבה יותר צנועות. המחברים גם שמו לב כי conjugation של מטריצה A במערכת הדינמית (הכפלתה מימין ומשמאל במטריצה אוניטרית V) מוביל למערכת דינמית שקולה עם התוצאה Vx. הבעיה שמטריצה A מ-HiPPO לא ניתן לתאר בצורה V\*LV כאשר L היא מטריצה אלכסונית, ו V היא מטריצה אוניטרית (נובע מכך ש A אינה קומוטטיבית עם A\* כלומר לא נורמלית - זה השם אין מה לעשות).

אז הכל אבוד? מתברר שלא. מתברר ש A מ HiPPO ניתן לתאר בתור סכום של מטריצה נורמלית ומטריצה בעלת רנק נמוך (עבור LegT הרנק אפילו שווה ל-1 כלומר תוספת זו כי מכפלה חיצונית של שני וקטורים בעלי מימד Nx1). ואז המאמר מציע אלגוריתם די לא טריוויאלי עבוד חישוב של קרנל קונבולוציה ארוך המבוסס על 3 עקרונות מתמטיים:

- במקום לחשב A^l עבור כל l ניתן לחשב z-transform (מקוטע עד L) של A ואז לחשב בצורה די פשוטה את A^l על ידי הצבה של שורש שונים של 1 (המרוכבים) ב z-transform הזה.

- כאשר A הוא הפרש של מטריצה אלכסונית L ומטריצה בעלת רנק נמוך מאוד ניתן לחשב את z-transform הזה בצורה יעילה דרך זהות Woodbury שמסתכם בהיפוך של מטריצה אלכסונית.

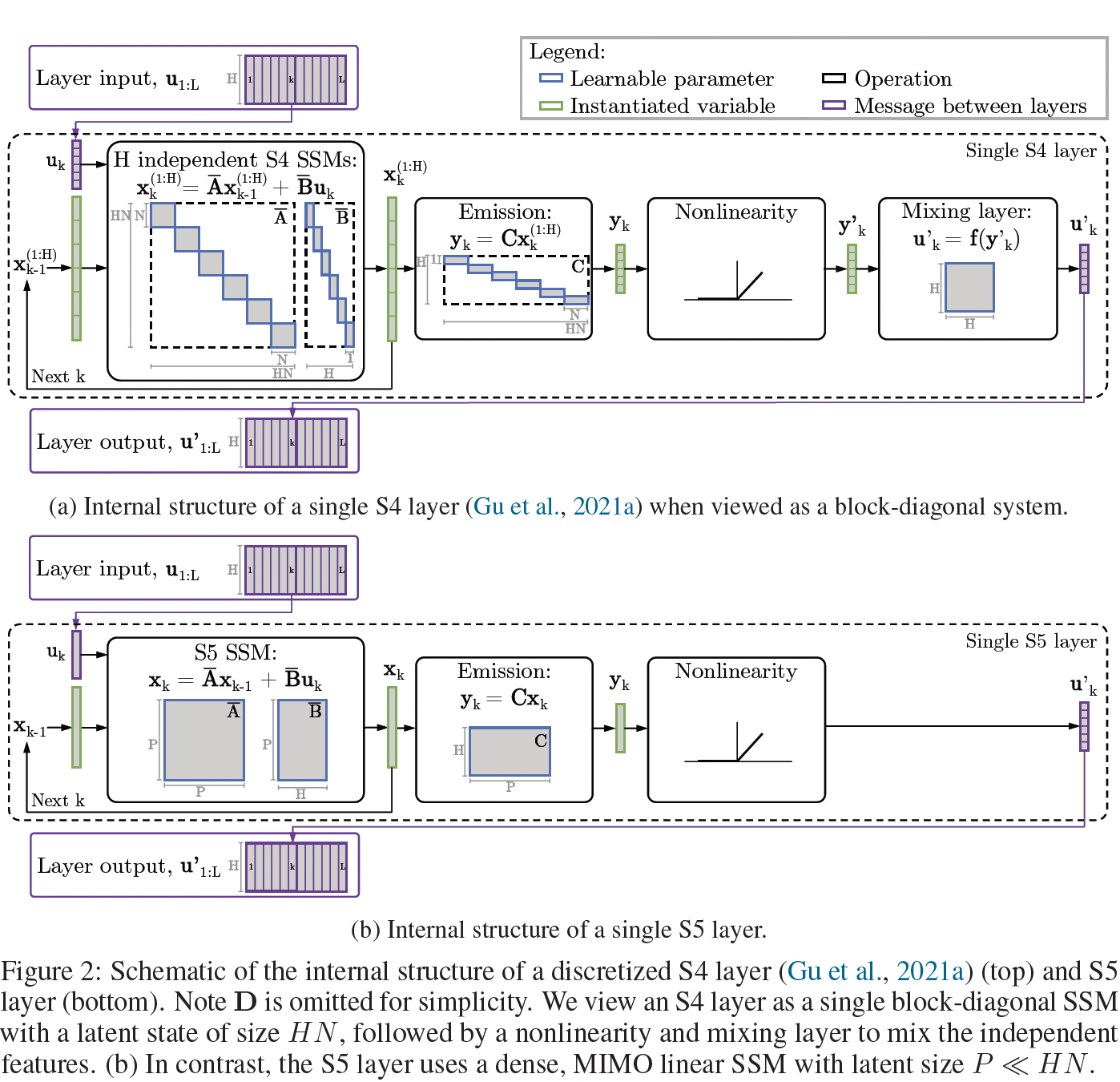
- ניתן לבצע את כל החישובים העולים כאשר מפעילים בזהות Woodbury בצורה יעילה מאוד עם Cauchy Kernel; שזה בגדול מטריצה שנבנית בצורה מסוימת משני וקטורים

לבסוף, מבצעים את החישובים האלו עבור כל מימד של ייצוגי הטוקנים בנפרד ואז מערבבים עם שכבה לינארית (או כמה). מטריצות אלכסוניות L (למעשה וקטור), וקטורים B, C וגם P ו-Q שמכפלתם היא מטריצה בעלת נמוך מאומנות בנפרד עבור כל מימד של ייצוג הטוקנים.

זהו, יצא ארוך - הסקירה הסקירה תהיה קצרה יותר.

## **Review 189: Simplified State Space Layers For Sequence Modeling(S5)**

https://arxiv.org/abs/2208.04933



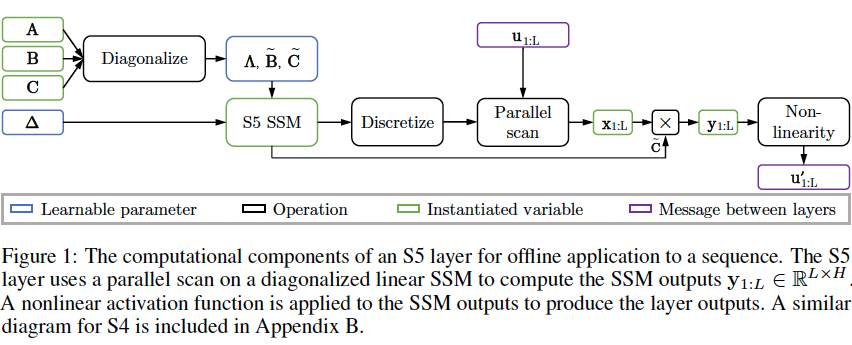
ממשיכים עם הסקירה החמישית כל הדרך לממבה. סקירה זו תהיה די קלילה כי היא בסך הכל מציעה שכלול לארכיטקטורת S4 שדיברנו עליה בהרחבה בסקירה הקודמת. למעשה S4 בנויה מ- H (מימד של ייצוג הטוקן) SSMs שכל אחד מהם מומש עם מערכת דינמית לינארית שדנו עליה בהרחבה בסקירות הקודמות. כל SSM מהווה בעצם זכרון עבור כל מימד של וקטור ייצוג הטוקן לאורך זמן. זמן כאן ציר הטוקנים שאותם אנחנו רוצים לזכור כדי לקבל החלטה מושכלת עבור הטוקן הנוכחי.

אם נביט בנוסחאות המתארות SSM ניתן לראות כי H מערכות SSM האלו אפשר לתאר כ-SSM אחד גדול המתואר על ידי מטריצה A בלוקית אלכסונית שבאלכסון שלה נמצאות מטריצות A\_i, i=1,...H המתארות כל SSM. וקטורים B ו- C של ה- SSM הגדול הזה ניתן לבנות על ידי השרשור של וקטורי B\_i ו- C\_i של H המערכות SSM האלה.

כמובן שכל הסיפור הזה דורש לא מעט זכרון ולא מעט חישובים במיוחד כאשר H (מימד ייצוג הדאטה) הוא סדר גודל של כמה מאות או כמה אלפים. אז המאמר המסוקר מציע להשתמש באותה מטריצה A עבור המערכות הדינמיות המתארות זיכרון של כל מימד שך ייצוג הדאטה. גודל של מטריצה A נבחר הרבה יותר קטן מ- PH שזה גודל של מטריצה A עבור כל המימדים של ייצוג התוכן יחד (= גודל המטריצה הבלוקים האלכסונית). כמובן שבדרך זו נחסכים לנו גם הזיכרון וגם כמות החישובים הנדרשת גם בהיסק וגם באימון.

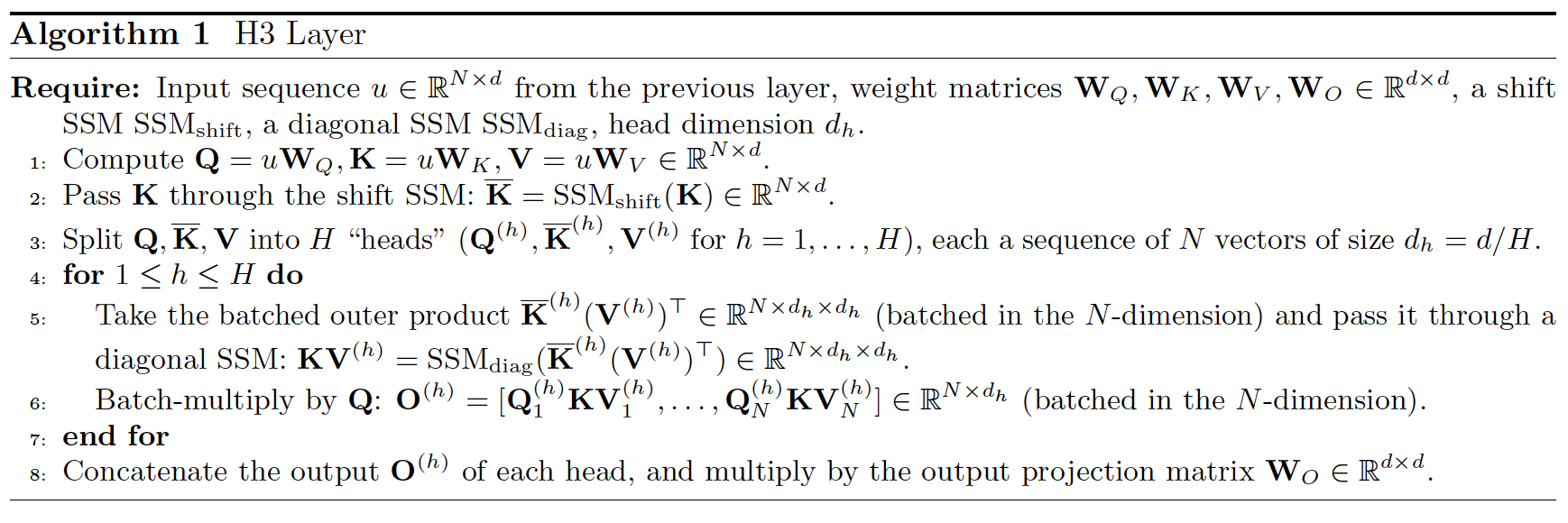
כמובן שהקטנה שכזו של מימד מטריצה A עלול לפגוע בביצועי המודל (כי אידיאלית זיכרון של מימדים שונים של ייצוג דאטה עשויים להכיל אופיינים שונים של זיכרון; נגיד, זיכרון ארוך וקצר טווח). המחברים בוחנים מספר דרכים לצמצום פגיעה זו על ידי עדכון חכם של A ועוד כמה טריקים נחמדים. המחברים למשל בוחרים אופצייה של מטריצה A בעלת מימד KP כאשר K הרבה יותר קטן מ-H.

5⃣בקיצור מאמר קליל וקל לקריאה…



## **Review 190: Hungry Hungry Hippos: Towards Language Modeling with State Space Models(H3)**

https://arxiv.org/abs/2212.14052



עד עכשיו ראינו מאמרים שמימשו את ארכיטקטורת SSM בתור רכיב הזכרון של המערכת. אף אחת מהמאמרים שסקרנו לא ניסה לשלב גישה זו(SSM) יחד עם מנגוננים אחרים שמוכרים לנו מעולם של עיבוד סדרות דאטה עם רשתות נוירונים. המאמר המסוקר משלב את גישת SSM, המיושמת באמצעות מערכות דינמיות לינאריות, עם מנגנון תשומת הלב הלינארי.

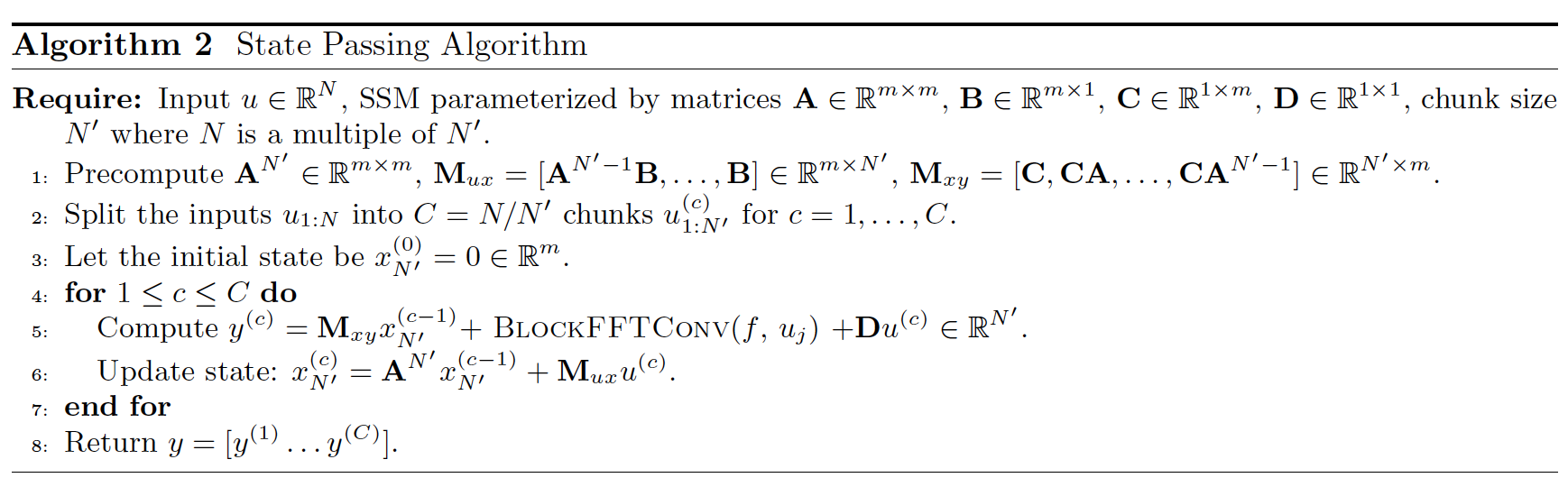
דיברנו על מנגנון attention הלינארי בסקירה השלישית של המאמר: Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention. המאמר הזה הציע להחליף את מנגנון תשומת הלב הרגיל עם softmax של הטרנספורמרים בחישוב לינארי: (f(k)\*f(q כאשר \* מסמן מכפלה פנימית ו- f היא פונקציה לא לינארית. המאמר מראה כי ניתן לתאר טרנספורמר עם מנגנון זה בתור RNN ולהימנע מסיבוכיות חישוב ריבועית הרגילה של הטרנספורמרים. כלומר אין צורך להתחשב בצורה מפורשת בכל פיסות הדאטה לפני טוקן i בשביל לחזות אותו אלא כל הזיכרון של הטוקנים הקודמים נדחס ושמור בשני וקטורים.

אוקיי, אבל למה צריך בעצם לשלב ארכיטקטורות מבוססת SSM עם מנגנונים אחרים? התשובה היא פשוטה - ארכיטקטורות אלה לא מספיק טובות לכמה משימות. למשל מחברי המאמר שמו לב כי במשימות כמו Induction Head שצריך לעקוב על טוקן שבא אחרי טוקן מסוים, ארכיטקטורה זו מפגינה ביצועים לא מרשימים במיוחד. כדי להתמודד עם סוגיה זו המחברים הציעו לשלב SSM עם מטריצות A מסוימות עם מנגנון תשומת הלב הלינארי.

אז איך כל הסיפור הזה עובד? בשלב הראשון מכפילים את ייצוגי הטוקנים במטריצות Q, K ו- V כמו בטרנספורמרים. בשלב השני מפעילים SSM על המפתח k (עבור כל הטוקנים) עם מטריצה A המדמה ״זיכרון של הטוקן הקודם״(בערך A\_ij=1 כאשר i - j=1 ואפס אחרת). מבחינת מנגנון תשומת הלב הלינארי זה ״מקביל״ ל (f(k למרות ש f כאן ״די לינארית״.

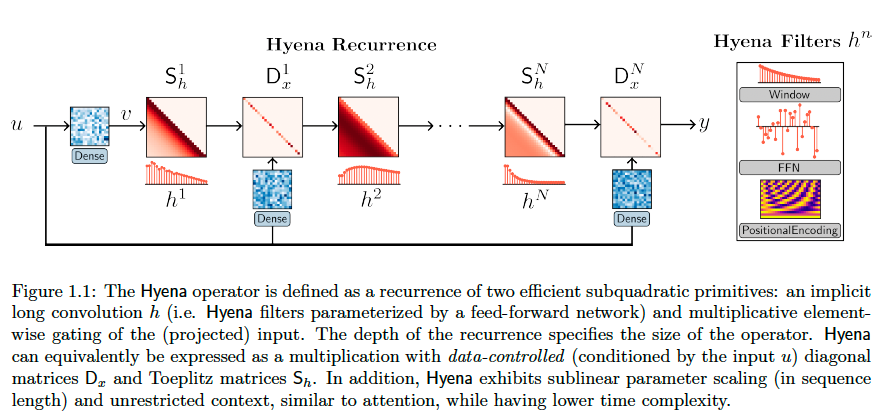
בשלב השלישי לוקחים q, v והתוצאה של השלב הקודם ל h חתיכות (= ״ראשים״ במנגנון ה-attention). לאחר מכן מכפילים כל חתיכה של q בחתיכה של התוצאה של השלב הקודם (עם k) ו״מעבירים״ את התוצאות דרך SSM עם מטריצה A אלכסונית. את התוצאה מכפילים ב-q, מאחדים את כל התוצאות ומכפילים במטריצה W\_O כמו שמקובל בטרנספורמרים מרובי ראשים(multi-head transformers).

בנוסף המאמר מציע מנגנון הנקרא FlashConv לחישוב חיזוי הטוקנים באופן מקבילי במהלך האימון. כמו שאתם זוכרים הקרנל קונבולוציוני שם מאוד ארוך וחישובו יכול להיות יקר גם מבחינת הזיכרון וגם מבחינת הזמן אם נעשה בצורה נאיבית. המחברים משכללים את המנגנון כאשר העיקרון המוביל הוא ניצול מקסימלי של זיכרון SRAM המהיר שיש ב-GPUs תוך מזעור של הערבות דאטה לשם (זה איטי ובד״כ מהווה צוואר בקבוק) . הזיכרון הזה לא גדול ולא ניתן לדחוף שם יותר מדי אז נדרשות שיטות מתוחכמות המפרקות את חישוב הקונבולוציה לחלקים תוך ניצול תכונות של FFT ו- IFFT. נזכיר שהחישוב הקונבולוציה מתבצע בצורה: ((c(x) = iFFT(FFT(c)\*FFT(x כאשר (c(x היא קונבולוציה על x עם קרנל c.



## **Review 191: Hyena Hierarchy: Towards Larger Convolutional Language Models**

https://arxiv.org/abs/2302.10866



היום סוקרים את המאמר השביעי בסדרה וכאן אני חייב להודות שלקח לי הרבה מאוד זמן לצלול למאמר הזה לעומק למרות שטכנית המאמר לא מורכב במיוחד (בטח לא קרוב ל Hippo). אבל המאמר כתוב בצורה נוראית: מצד אחד הוא עמוס בפרטים לא מהותיים ומצד שני נעשה מאמץ ניכר (על ידי המחברים) להסתיר את הפרטים המהותיים עם מלל אינסופי. לא יודע האם זה נעשה בזדון או לא אבל המאמר הזה לקח לי בערך פי 4 יותר זמן ממאמר ממוצע שזה הרבה סטיות תקן מהממוצע (יש לי מדגם די גדול).

אחרי ששחררתי את הקיטור אפשר להתחיל לסקור את המאמר זה שמציע הכללה חמודה ל H3 שסקרנו קודם. H3 היה די נחמד אבל עדיין הביצועים שלו לא היו בשמיים עבור כמה משימות על הדאטה בעלי אורך הקשר ארוך מאוד. אז באו לנו מחברי Hyena והציעו לשפר את ביצועי H3 אך לא במחיר של עלייה ניכרת במשאבי חישוב והזיכרון.

אוקיי, אז מה הם הציעו בעצם? אתם זוכרים שב-H3 אנו לקחנו וקטורי מפתח עבור הטוקנים בתוך חלון ההקשר (=מטריצה K) העברנו אותם דרך SSM (State-Space Models) ואז הכפלנו אותם בווקטורי שאילתה (=מטריצה Q) והעברנו את התוצאה דרך SSM נוסף עם מטריצה A אחרת ואת התוצאה הכפלנו בווקטורי ערך עבור כל הטוקנים בתוך חלון ההקשר (=מטריצה V)? כל המנגנון הזה הוא למעשה attention לינארי.

אז ההכללה הראשונה המוצעת במאמר היא הגדלת מספר הוקטורים שעליהם מופעלת SSM (בצורה לא מפורשת - נדבר על זה עוד מעט) ל N. כלומר יש לנו 1+N הטלות של ייצוג הטוקנים (אחת עבור מטריצת הערך V). אחרי שיש לנו את ההטלות האלו מפעילים עליהם מה שבמאמר נקרא Short Convolution (קונבולוציה קצרה) בציר הטוקנים. זה נעשה כנראה כדי ללמוד את האינטראקציות בין הטוקנים הסמוכים (המאמר לא מסביר כלום לגבי זה).

מפה העניינים קצת מסתבכים. אנו לוקחים מטריצת הערך V מההטלה האחרונה ומפעילים עליהם SSM (אותה מערכת דינמית לינארית) אבל בצורה לא מפורשת. מה זה אומר אבל? אנו יודעים שהפעלת SSM לסדרה של L טוקנים שקולה להפעלה של קרנל קונבולוציה באורך L על ייצוגי טוקנים אלו. קרנל קונבולוציה זה מוגדר על ידי המטריצות המגדירות את ה-SSM (שזה A, B, C). אז ניתן להגדיר SSM בצורה לא מפורשת דרך הקרנל הזה. צריך לזכור פעולה זו שקולה להכפלת וקטורים, המרכיבים מטריצת ערך V, במטריצת קונבולוציה גדולה (= שזה אותו מנגנון של attention לינארי).

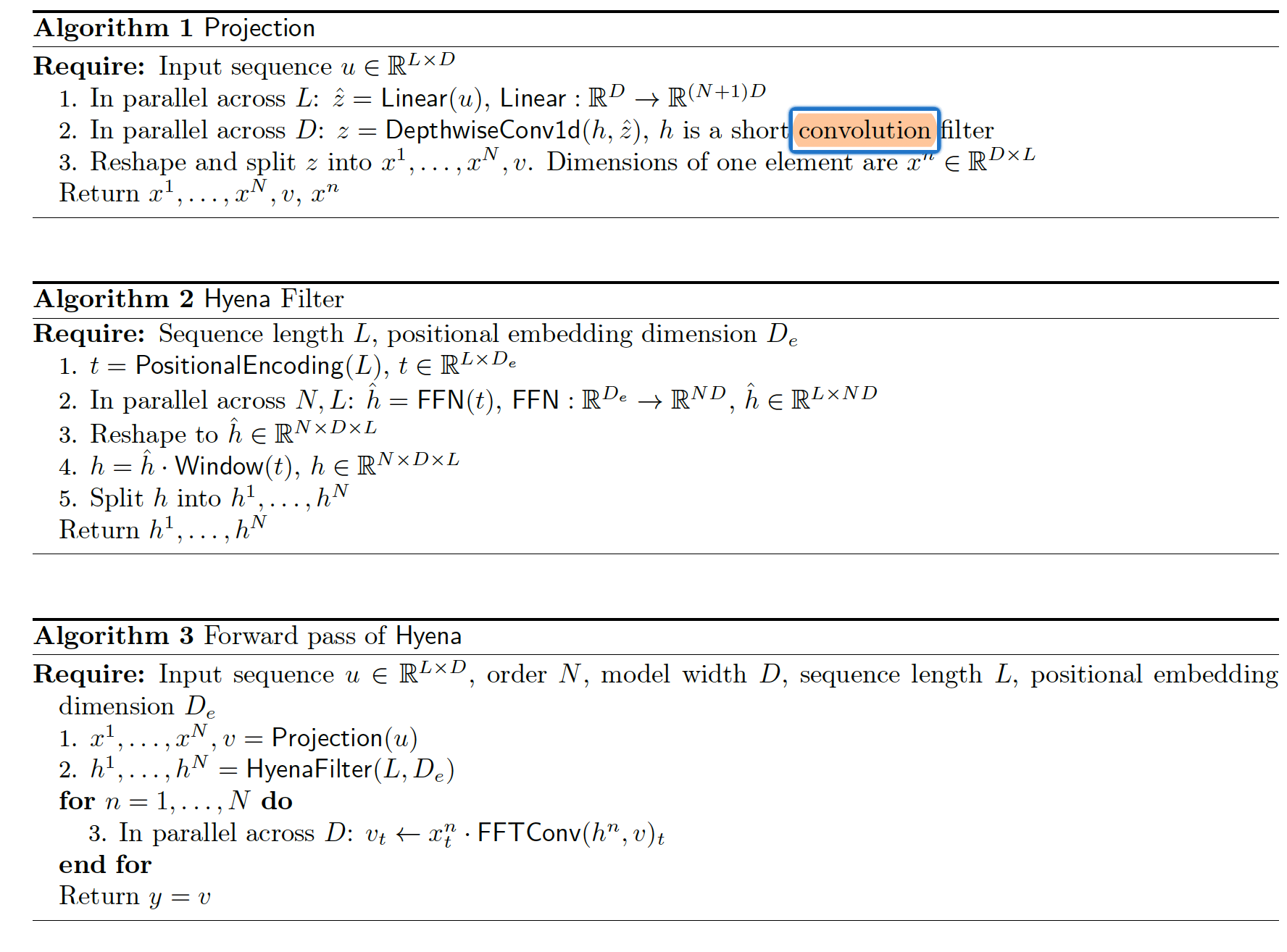
למשל ב-H3 (שסקרנו בפעם הקודמת) היו לנו שני SSMs (עם מטריצה אלכסונית ועם מטריצת הזזה ב-1) ומתברר שניתן לייצג אותם בצורה לא מפורשת עם קרנל שהוא מכפלה של שתי מטריצות שכל אחת מהן היא מכפלה של מטריצה אלכסונית במטריצת Toeplitz. מה שמיוחד במטריצת Toeplitz היא שכל שורה בה כי הזזה שמאלה של השורה הקודמת. תכונה מעניינת של כל מטריצה Toeplitz היא שהיא מהווה ייצוג של קרנל קונבולוציה.

אז המחברים לקחו את הייצוג הלא מפורש של SSM ובנו אותו מ- N מכפלות של מטריצות אלכסוניות ומטריצות Toeplitz (שונות). כלומר מתחילים מטריצה V עבור הטוקנים מפעילים עליה מיפוי $H$ לינארי (= קרנל קונבולוציה) די מסורבל. כלומר H הוא הרכבה של N מיפויים $H\_i, i=1,...N$ לינאריים שכל אחת מהן היא קונבולוציה המיוצגת על ידי מטריצה Toeplitz (מס' i) ומכפלת התוצאה איבר-איבר בהטלה מספר i של וקטורי הטוקנים. במאמר כל הסיפור הזה נקרא Hyena operator מסדר N.

אוקיי, מה הבעיה העיקרית עם הגישה הזה? זה דורש הרבה זיכרון בטח עבור N גדול יחסית. אז המאמר מציע פתרון מאוד אלגנטי. במקום ללמוד את כל N קרנלים אלו בצורה מפורשת נגדיר אותם באמצעות רשת נוירונים רדודה (fully-connected). גם נוכל לשלוט על מספר פרמטרים וכך לשמור על זיכרון קבוע פחות או יותר לכל ערך של N. כך ניצור את כל N קרנלים עם רשת אחת בלבד. ארכיטקטורת רשת רדודה זאת היא די מיוחדת והיא מכילה פונקציות אקטיבציה מחזוריות (כדי ליצור קונבולוציות עם תדרים גבוהים).

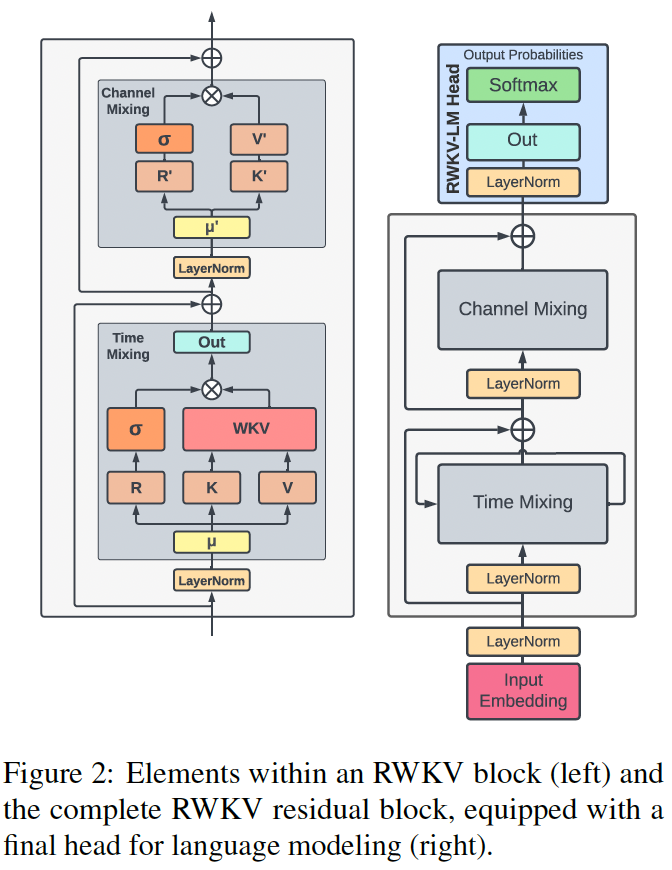
בנוסף מכפילים קרנל זה (איבר איבר) בפונקציה מעריכית עם פרמטר חיובי דלטה $\exp(-\delta t)$ בציר הטוקנים. הכפלה זו באה לשקף דעיכה בהתחבשות בטוקנים(=attention( ככל המרחק בינם לבין הטוקן החזוי יורד. המאמר משתמש בכמה אופרטורי Hyena (ערוצים) במקביל עם מקדמי $delta$ שונים המבטאים קצבי דעיכה שונים של attention. כל אופרטור כזה מופעל על וקטורי קידוד מיקומי (positional encoding).

ודבר אחרון: כל הקונבולוציות מחושבות דרך FFT(Fast Fourier Transform) וגם IFFT כמו במאמר של H3 (כי זה פשוט יותר מהיר). כמובן כל SSM (גם לא מפורש) מופעל על כל מימד של ייצוג הטוקנים שטיפה מסבך את התיאור אבל עדיין הכל נשאר לינארי.



## **Review 192: RWKV: Reinventing RNNs for the Transformer Era**

https://arxiv.org/abs/2305.13048



אוקיי, אחרי כמה מאמרים כבדים הפעם יש לנו מאמר קליל יחסית. אתם אולי זוכרים שהמאמר השלישי שסקרנו בסדרה (״Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention״) הראה שטרנספורמר עם attention לינארי ניתן לייצג בתור RNN מצד אחד (כלומר ניתן להפעלה באופן איטרטיבי כאשר הוא דוחס את הטוקנים הקודמים בוקטור זיכרון אחד) ומצד שני ניתן להפעלה באופן כמו הטרנספורמר מן המניין. כלומר יש בו את הדואליות שרצינו: חיזוי מקבילי של טוקנים ממוסכים במהלך האימון וחיזוי טוקנים בעל סיבוכיות לינארית במהלך ההיסק (inference).

המאמר שנסקור היום מקרב את הטרנספורמר ו-RNN באופן מפורש אפילו קצת יותר. המחברים לוקחים טרנספורמר עם מנגנון ״attention״ ״פשוט יותר״ ומוסיפים קצת RNN לאופן בו מחושבים מטריצות מפתח K ומטריצת ערך V. אבל קודם אספק לכם כמה פרטים על מנגנון "attention" שלקחו המחברים בתור בסיס ולמה אני שם אותו כאן בגרשיים. אז מנגנון הזה נלקח מהמאמר AFT) An Attention Free Transformer) שלפי שמו נראה שהמאמר מציע טרנספורמר ללא attention כלל!

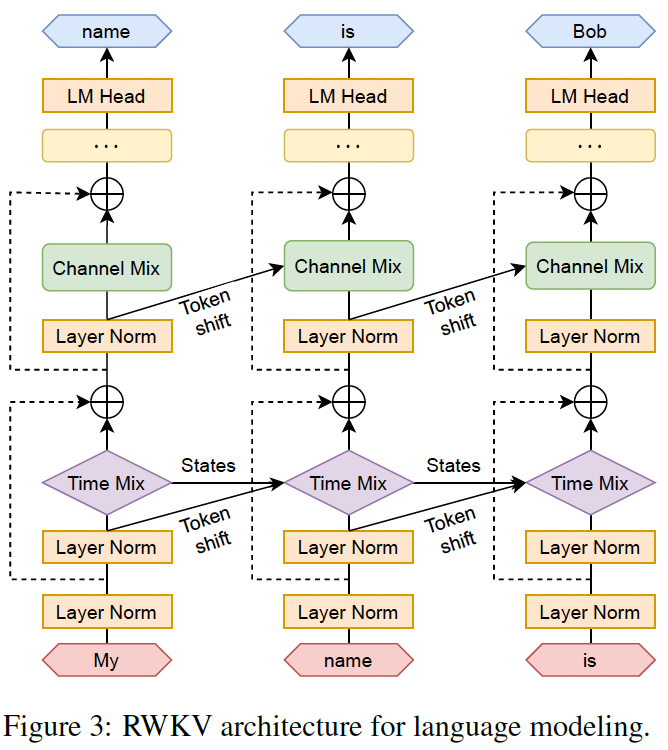
אוקי, אז מה הסיפור של AFT ומה זה בכלל טרנספורמר ללא attention (לי זה נשמע על ההתחלה כמו אוטו ללא מנוע). AFT מחליף את המנגנון הרגיל של חישוב attention של הטרנספורמר בכזה שדורש משמעותית פחות זיכרון מהטרנספורמר הרגיל (בגרסתו הפשוטה גם סיבוכיות חישובית מוקטנת עד כדי לינארית במונחי אורך הקלט) ועושה את זה בדרך מאוד הגיונית. AFT מחליף את המכפלות הפנימיות בין וקטור שאילתה q\_i ווקטור המפתח k\_j באקספוננט של סופטמקס (שזה הלב של המנגנון והסיבה לסיבוכיות הריבועיות) בסכום של וקטורי המפתח עם מטריצת משקלים נלמדת w\_ij (מנורמל). כלומר לא מתחשבים בוקטורי שאילתה q\_i אלא **משתמשים במקדמים קבועים ומחושבים על סמך סט האימון**. לאחר מכן בונים צירוף לינארי עם וקטור הערך v כמו בטרנספורמר הרגיל.

כלומר מקדמי ה-attention בין טוקן i לטוקן j לא תלויים באופן מפורש בייצוג טוקן i אלא רק ב- i ו-j. בחירות חכמות (פרמטריזציה) של w\_ij מאפשרות להקטין את דרישות זיכרון והסיבוכיות החישובים כאשר המחיר הוא כמובן expressiveness של המודל. אחת הבחירות של w\_ij היא פונקצית דועכת מעריכית כאשר הארגומנט הוא מרחק בין הטוקנים (המאמר המסוקר משתמש בה).

אוקיי, אז איך מלבישים על זה RNN? לוקחים את המנגנון ה- attention מהפסקה הקודמת עם שפצור קל ליציבות נומרית - הוספה של וקטור u (המנגנון הנקרא wkv) ומפעילים אותו עם וקטורי מפתח וערך K ו- V מחושבים כמו ב-RNN. כלומר בונים וקטורים אלו( K ו- V) תלוים באופן מפורש בייצוג הטוקן הנוכחי ו**גם בייצוג הטוקן הקודם(**זה כל הקטע).במקום להכפיל את ייצוג הטוקן במטריצות W\_k ו- W\_v (כמו בטרנספומר הרגיל) מכפילים אותם בסכום ממושקל (עם משקלים נלמדים) של ייצוג הטוקן x\_t הנוכחי וייצוג בטוקן הקודם {x\_{t-1. בנוסף מחשבים וקטורי r (הנקרא receptaince) באותה הצורה (עם x\_t ו- {x\_{t-1 ומטריצת W\_r). וקטורי r למעשה משמשים לנו כדי ״לשערך״ עד כמה אנו צריכים להתחשב בה (מחושבת עם הסיגמואיד כמו בזמנים הטובים ב-RNN). כל הסיפור הזה נקרא באופן לא מפתיע rkwv.

בסוף משלבים את התוצאה של rwkv עם וקטורי מפתח וערך המחושבים באותה צורה כמו ב-rwkv (התחשבות ב- x\_t ו- {x\_{t-1 אבל עם מטריצות הטלה נלמדות אחרות). איך משלבים? כרגיל בצורה של ResNet.

וזהו זה. שמח לבשר שהמאמר הבא שנסקור בדרך לממבה גם יהיה קליל (Retentive Network).



## **Review 193: Retentive Network: A Successor to Transformer for Large Language Models**

https://arxiv.org/abs/2307.08621

זה הולכת להיות הסקירה הקלה ביותר (אך קצת ארוכה). המאמר משתמש באופן די אלגנטי ברעיונות שהוצע ב 8 המאמרים שכבר סקרנו. אזכיר שהמכנה המשותף במאמרים שסקרנו היתה מטרה למצוא ארכיטקטורה בעלת דואליות הבאה:

⬅1. ניתנת לאימון באופן מקבילי כמו הטרנספורמרים

⬅2. היסק (inference) מהיר (=לינארי במונחי אורך חלון הקשר) שלא מצריך התחשבות מפורשת בכל טוקני של חלון ההקשר.

הארכיטקטורה שהמאמר מציע היא אכן מבורכת בדואליות זאת ובאותו הזמן היא מאוד פשוטה וקלה להסבר (ככה נראה לי). אתם בטח זוכרים את הייצוג הקונבולוציוני של (SSM (state-space model עבור ייצוג הזיכרון של סדרת טוקנים?

אם לא אזכיר בקצרה. עבור סדרת טוקנים נתונה יש לנו מערכת דינמית לינארית (DMS) שבאמצעותה אנו מייצגים בצורה איטרטיבית את זיכרון s\_n הנצבר ב n הטוקנים הראשונים בסדרה. בעזרת DMS ניתן לחשב את s\_n מייצוג הזיכרון קודם {s\_{n-1 ומייצוג של טוקן ה-n, מסומן v\_n.

לאחר מכן באמצעות וקטור s\_n אנו ממדלים פלט המודל o\_n עבור טוקן n (= ייצוג תלוי הקשר או contextualized embedding של טוקן n) דרך הטלתו עם מטריצה Q. נציין כי DMS מגדירה את מעבר(הלינארי) בין ייצוג של הזיכרונות n-1 ו-n מאפשר חיזוי במקבילי עבור כמה טוקנים במהלך אימון.

אותה DMS מוגדרת באמצעות מטריצות A ו-K וכאמור הפלט o\_n מוגדר באמצעות מטריצה הטלה Q. מטריצות Q ו- K הן אלו שנקראות בטרנספורמר מטריצות שאילתה וערך ומחושבות באותה צורה: Q = X\*W\_Q, K = X\* W\_K, כאשר X הוא ייצוגי הטוקנים.

עכשיו השאלה איך אנו מגדירים חישוב מקבילי של o\_n עבור כמה n? הרי עבור n גדול מספיק העלאה של מטריצה A בחזקה עלולה להיות יקרה גם מבחינת זיכרון וגם מבחינת מאשבי חישוב. אז פותחים אחד הפרקים הראשונים של ספר של אלגברה לינארית ומגלים שניתן לתאר מטריצות ריבועיות (לא כולן!) בתור A=L\*D\*L^{-1} כאשר D היא אלכסונית עם ערכים מרוכבים lambda\_j \* exp(i\*theta\_j), j=1,....,d.

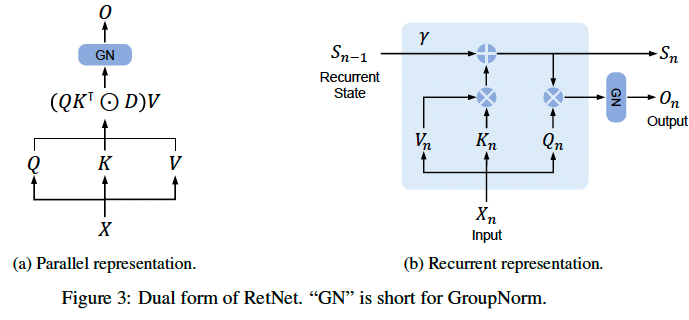
מה בעצם טוב בייצוג הנחמד הזה? זה מאפשר לנו להעלות את מטריצה A בחזקה והבעיה שלנו עם חישוב A^n נראית פתורה. המאמר גם מניח ש lamda\_j = lambda, j=1,..,d וזה מאפשר את הייצוג הבא של המודל שהם מציעים:

למעשה המחברים מחליפים את מנגנון ה-attention הממומש עם סופטמקס בטרנספורמרים עם ה-attention הדועך בצורה מעריכית כפונקציה של בין הטוקנים. חדי העין שקראו את הסקירה הקודמת שלי ישימו לב שעיקרון דומה ממומש גם ב-RWKV אבל די מיצוע מעריכי של המידע מהטוקן הקודם. וכמובן ייצוג כזה חישוב מהיר עבור כל טוקן במהלך היסק (שזה תכונה 2 שלנו).

המאמר מציע שני שכלולים נחמדים ל-RetNet. הראשון הוא כדי להאיץ את מהירות האימון עוד יותר ולנצל את משאבי החישוב הזמינים ניתן לחלק את הטוקן לצ'אנקים ולהפעיל חישוב מקבילי בתוך כל צ'אנק וחישוב איטרטיבי בין צ'אנקים.

שכלול נוסף הוא שימוש במקדמי gamma שונים ל״ראשים״ (heads) שונים של RetNet. זה למעשה מקנה למודל יכולת יותר להתמקד בטוקנים קרובים יותר (lambda גבוה) ו״לפזר״ את ה-attention גם טוקנים רחוקים(lambda נמוך). שילוב של ראשים בעלי lambdas שונים ״לחקות״ את הטרנספורמר (לפחות במידה מסוימת).

https://arxiv.org/abs/2307.08621



## **Review 194: Mamba: Linear-Time Sequence Modeling with Selective State Spaces**

<https://arxiv.org/abs/2312.00752>

זה קורה עכשיו, אחרי 9 סקירות שחלקם היו די לא פשוטות הגענו למטרתנו הקדושה שזה Mamba. מכיוון שאני מפרסם סקירות בשלשות (באתר MDLI) אני אוציא עוד 2 סקירות נוספות של שכלולי ממבה (אחד מהם Mamba MoE והשני עוד הוחלט).

האמת שאחרי שאנו הבנו מה- (SSM (space-state models ואיך ניתן לבנות ארכיטקטורה מבוססת עליהם לעיבוד דאטה סדרתי, השכלול המוצע על ידי mamba הוא די אינטואיטיבי ומתבקש. כמו שאתם זוכרים SSM ממומשת בתור מערכת דינמית(DLS) לינארית כאשר הקלט למערכת זו היא ייצוג וקטורי (embeddings) של איברי הסדרה (= טוקנים).

בשלב הראשון המערכת הדינמית מחשבת וקטור **s** הוא הוא ייצוג דחוס של זיכרון כלומר וקטור ״הזוכר״ את המידע הרלוונטי עבור כל הטוקנים הקודמים לטוקן הנוכחי. בשלב השני מחשבים את הפלט עבור טוקן זה המוזן לשכבה הבאה (שיכולה להיות גם שכבת שמייצרת פלט סופי). כל חישובים אלו מתבצעים באמצעות מיפויים לינאריים כלומר מכפלות במטריצות. חשוב להבין שכל המעברים בין ייצוגי הזכרון בין הטוקנים הם לינאריים ונשלטים על ידי אותה מטריצה A ווקטורי B, C וסקלר delta. ד״א פרמטר delta מגדיר (באופן גס) את קצב דעיכה של הזכרון (כלומר ככל ש delta גבוה יותר אנו נוטים ״לזכור״ פחות מהטוקנים הקודמים).

מה היתרונות של הארכיטקטורה הזו? היא בעלת תכונה הדואליות המיוחלת המשלבת 2 התכונות הבאות:

* ניתן לחזות באופן מקבילי (בו זמנית) כמה טוקנים במהלך אימון (כמו בטרנספורמרים)
* חיזוי מהיר של טוקן במהלך היסק (ללא התחשבות בכל הטוקנים בחלון ההקשר כמו בטרנספורמרים שמביא לנו את הסיבוכיות הריבועית).

כלומר הארכיטקטורות מסוג זה הם יעילות בזמן האימון ומהירות בזמן ההיסק. אבל כמו שאתם יכולים לנחש יש לנו מחיר לשלם על כל התכונות הנחמדות האלו. ומחיר הוא כמובן יכולת של המודל למדל תלויות מורכבות של הדאטה. עקב כך מאמרים כמו Hyena, H3, S4 ניסו ניסו לבנות את הפרמטרים של DLS (המגדירה מעברים בין ייצוגי הזכרון ויצירת הפלט) בצורה חכמה (ודי מורכבת).

אבל מתברר שזה לא מספיק. מעברים לינאריים עם פרמטרי DLS קבועים לא מסוגלת למדל דאטה מורכב (כמו שפה טבעית). אחד המשימות שמודל כזה נכשל עליה הוא העתקת טוקנים הבאים אחרי טוקן ספציפי (וזה די הגיוני לאור הפרמטרים הקבועים של DLS). כמו שאתם יכולים כבר לנחש אולי מחברי ממבה מציעים לעשות חלק מהפרמטרים (B, C ו delta) תלוים בייצוג הטוקן הנוכחי. התלות הזו היא לינארית עם מטריצות נלמדות. וזה עוזר לנו להתחשב בפיסת הקלט הנוכחית בצורה יותר טובה. כאמור B, C מגדירות את האופן בו ייצוג הזכרון והפלט עבור הטוקן הנוכחי בהתאמה ואז יש לנו סיכוי יותר טוב להצליח במשימות מהסוג שתיארתי לפני. בנוסף תלות של delta בייצוג הטוקן הנוכחי מקנה לנו אפשרות לשחק עם קצב דעיכה בצורה יותר גרנולרית שמקנה לנו יכולת ״לשכוח״ ו״לזכור״ איפה שצריך.

אבל האם איבדנו את הדואליות שלנו בדרך. מתברר שלא, הרי המעבר בין ייצוגי הזיכרון של הטוקנים עדיין לא תלוי במיקום של הטוקן אלא בייצוג. כלומר אנו עדיין יכולים לחזות מספר טוקנים בו זמנית כי אנו יכולים לחשב את כל הפרמטרים מראש (לא צריך לחשב את המצב הזכרון הקודם באופן מפורש). וכמובן אין צורך להתחשב בכל הטוקנים בתוך חלון ההקשר במהלך ההיסק כי הזיכרון עדיין מיוצג על יד וקטור אחר. אז יש דואליות!

מה שכן קורה הוא זה החישובים הופכים לקצת יותר מורכבים (שימו לב שהחלק הכי בעייתי בחישוב שהוא העלאה של מטריצה בחזקה לא השתנה כי A נותרה קבועה). בסוף המאמר מציע כמה שכלולים לאופן חישוב המייעלים ומזרזים אותו (המשחקים בין זכרון מהיר ואיטי של GPU).

זה וזה עכשיו אתם יודעים מה זה ממבה. נתראה ב Mamba MoE עוד כמה ימים.

## **Review 195: Can Mamba Learn How to Learn? A Comparative Study on In-Context Learning Tasks**

https://arxiv.org/abs/2402.04248

אוקיי, סוקרים מאמר הבא בסדרת ממבה (מה שבא אחרי). בניגוד להצהרותיי בסוף סקירתי הקודמת לא תהיי Mamba MoE אלא מאמר אחר. הסיבה היא שלדעתי כמות המאמרים על MoE היא גדולה מאוד והמאמר הזה רק מציע להלביש אותו על Mamba ללא חידושים מעניינים אחרים אז החלטתי לדלג.

המאמר שנסקור היום בודק את האם מודלים המבוססים על ארכיטקטורת ממבה על למידת in-context (או ICL). למעשה ICL היא יכולת של מודל לבצע למידה על בסיס כמה דוגמאות בודדות (גם נקרא למידת few-shot) ללא שינוי של משקלי המודל. בגדול יכולת זו של הטרנספורמרים לא מאוד מפתיעה כי ה״חיזויים״ שלהם תלויים ביחסים בין חלקי הדאטה השונים(טוקנים) באופן מפורש באמצעות מנגנון ה-attention שלהם. כמובן יש מחקרים לא מעטים ומעניינים שחוקרים את התופעה המרתקת הזו ואני ממליץ לכם בחום להעיף מבט.

לעומת זאת הארכיטקטורה של ממבה לא לוקחת את היחסים בין הטוקנים השונים של הדאטה באופן מפורש ודוחסת את ה״עבר״ בוקטור אחד אז היכולת שלה לבצע ICL היא פחות אינטואיטיבי. זה אכן פחות קורה. המאמר בדק כמה ארכיטקטורות מבוססות (SSM (state-space models כמו ממבה, S4 וגם S4-mamba ללא מנגנון attention של הטרנספורמרים והשוו את יכולות ICL שלהם עם ארכיטקטורות הברידיות: כלומר שילוב של ממבה עם מנגנון ה-attention של הטרנספורמרים.

איך משלבים ממבה עם הטרנספורמר? המאמר בדק שתי גישות (די דומות). בגישה הראשונה הוא החליף את MLP שיש בבלוקי טרנספורמר אחרי ה-attention במנגנון של ממבה. הגישה השנייה (הכי מוצלחת) הנקראת MambaFormer בנוסף מחליפה את הקידוד המיקומי (positional encoding) של עוברים הטוקנים בממבה נוספת.

כאמור MambaFormer הגיע לביצועים הטובים ביותר מכל הארכיטקטורות הלא היברידיות (הטרנספורמר הטהור וכמה וריאנטים של SSM) באופן לא מפתיע בכלל. הרי MLP (רק 2 שכבות) ממדלים הפעולה די פשוטה ו-mamba היא למעשה מנגנון של זכרון הדוחס את המידע המהותי (בתקווה) של העבר (בטוקנים הקודמים). לא פלא שזה ניצח את כולם.

נשאר לנו רק לציין איזה משימות ניתנו למודלים אלו כדי לבחון את יכולות ICL שלהם. אחת המשימות היא לתת למודל כמה זוגות של (x, f(x)) עבור פונקציה f לינארית ולבקש ממנו לחשב (f(x עבור x-ים נוספים. משימה אחרת היתה לתת לה נקודות שנדגמו מ Gaussian Mixture מסוים ולבקש ממנו לדגום עוד נקודות. טבלה עם כל המשימות מצורפת לפוסט.

נתראה בסקירה ממבה הבאה והאחרונה (לא בחרתי עדיין).

## **Review 196: VMamba: Visual State Space Model**

<https://arxiv.org/abs/2401.10166>

מתחילים את הסקירה האחרונה בסדרת מאמרי ממבה. באופן די טבעי המאמר הזה מדבר על שילוב של ארכיטקטורה זה למודלי הראייה הממוחשבת (או ויז'ן בקצרה). הסקירה הולכת להיות די קצרה וקלילה.

הפעם לא אספק לכם סקירה על ארכיטקטורות מבוססת SSM (הייתי זאת לפחות ב 3 הסקירות הקודמות). כמו שאתם יודעים הטרנספורמרים השתלטו היום גם על תחום הויז'ן והחלק הארי של מודלי SOTA בתחום מבוסס על הטרנספורמרים. הטרנספורמרים החליפו את רשתות קונבולוציה(CNN) ששלטו בתחום הויז'ן עד 2020 בערך. למרות שיש טוענים שבכל מודל ויז'ן עובד יש או איזה backbone מבוסס CNNs או שמכיל רכיב כמו attention לוקאלי (דרך להתחשב ב-inductive bias שיש בדאטה ויזואלי שאותה מנצלים CNNs), עדיין השליטה של הטרנספורמרים בויז'ן נראית די מוחלטת.

אוקיי, אתה זוכרים שהמטרה של ממבה שהמודלים שקדמו לה היה הדואליות(ראו הסבר מפורט בסקירות הקודמות) המאפשרת אימון מקבילי לצד היסק (inference) מהיר. בגדול מחליפים את מנגנון ה-attention שיש בטרנספורמרים במודל מבוסס SSM שמאפשר דחיסה של כל הזכרון(עד טוקן הנחזה) בוקטור אחד ובכך מאפשרים היסק מהיר (וגם אימון מקבילי).

אז למה לא נעשה את אותו הדבר עבור הדאטה הויזואלי? זה בדיוק מה שהמאמר מנסה לעשות. למעשה המאמר משלב CNNs (הרי נפטרים מהטרנספומרים) עם מנגנון דחיסת הזכרון המבוסס SSM. מה שקצת משעשע שבמהלך האימון מודלי בסגנון ממבה מופעלים דרך קונבולוציה ארוכה (הוסבר בסקירות הקודמות בהרחבה) אז קיבלנו בסוף רשת קונבולוציה טהורה (לפחות באימון).

מה הבעיה העיקרית עם הכנסה של קצת -SSM -ים למודלים ויזואליים? הכיוון!! הרי עבור שפה טבעית וגם עבור אודיו די ברור שעבור טוקן נתון צריך ״לזכור״ את הטוקנים מתחילת הטקסט/אודיו. בתמונה לטוקן נתון (פאץ') ניתן לבנות זיכרון מכיוונים שונים (הסדר חשוב בממבה וב-SSM-ים אחרים). אפשר להתחיל מלמעלה, או מלמטה שלה תמונה, ללכת ימינה או שמאלה. לא ברור מה הכי טוב מבחינת ביצועים ואז VMamba משלב אותם. לפאץ' נתון מתחילים מפאץ' השמאלי והעליון והפאצ'ים נכנסים ל-SSM בכיוון ימין-מטה. משלבים את ה-SSM הזה עם ה-SSM שנבנה החל מהפאץ' התחתון מימין כאשר כאשר הפאצ'ים נכנסים ל-SSM מכיוון שמאל-למעלה. ככה בונים את החלק ה-SSM של VMamba. מעניין למרות שהמאמר נקרא ממבה הוא טוען שהוא משתמש בארכיטקטורה של S6 שקדמה לממבה (ההבדל הוא תלות של מטריצות B ו- C של SSM בייצוג הטוקן הנוכחי).

שאר הדברים הדי סטנדרטיים: חלוקת תמונה לפאצ'ים, הפעלה כמה סבבי downsampling המבוצע עם בלוקים המכילים קונבולוציות 3x3, SSM, חיבור resnet וכמה שכבות לינאריות.

זהו זה, סוף הסקירה מקווה שנהנתם לקרוא את סדרת ממבה…

## **Review 197, Short: LLM4Decompile: Decompiling Binary Code with Large Language Models**

https://arxiv.org/abs/2403.05286

המאמר מציע LLM4Decompile, משפחה של מודלי LLM לדיקומפילציה בגישה פתוחה שנעים מ-1B עד 33B פרמטרים. מודלים אלו מאומנים על 4 מיליארד טוקנים של קוד מקור בשפת C וקוד אסמבלי מתאים. המחברים גם מציגים את Decompile-Eval, הדאטהסט להערכת דיוק הדקומפילציה המבוצעת על ידי מודל (מקמפלים מחדש ובודקים את הפונקציונליות של הקוד).

LLM4Decompile מצליח לבצע דקומפילציה בצורה מדויקת 21% מקוד האסמבלי, עם שיפור של 50% ביחס ל-GPT-4. מודלי שפה לקימפול ולדקמפול של קוד נראים כמו תחום מחקר חשוב במיוחד לבניית סוכני AI חסינים יותר נגד התקפות אדוורסריות שיפעלו בשכבות העמוקות יותר של Software Stack.

## Review 198: Improving Text Embeddings with Large Language Models

<https://arxiv.org/abs/2401.00368>

הסקירה הזו תהיה לא סטנדרטית ואתחיל אותה משאלה: למה בחרתי לסקור את המאמר הזה?

לא בגלל שמדובר במודלי שפה - הרי כל יום יוצאים עשרות מאמרים על LLMs. גם לא בגלל שהמאמר הזה מציע שיטה לשיפור ביצועים של פתרונות (RAG (Retrieval Augmented Generation. הסיבה האמיתית היא שימוש בטכניקה שאני מאוד אוהב הנקראת (Contrastive Learning (CI או למידה ניגודית בשפת הקודש.

בזמן האחרון אני לצערי לא רואה יותר מדי עבודות שמשתמשות בפרדיגמה היפה הזו שתיארתי אותה בהרחבה בלא מעט סקירות בשנים האחרונות. בד״כ משתמשים בגישות השונות של הלמידה הניגודית כדי להפיק ייצוגי דאטה עוצמתיים במרחב בעל מימד נמוך או embeddings. ייצוג דאטה עוצמתי - הוא ייצוג שמצליח ״לשמור״ על פיצ'רים האינהרנטיים של פיסת דאטה, כלומר כזה שדוחס את הדאטה בצורה יעילה.

איך השיטה הזו עובדת? העיקרון הוא די פשוט - לכל פיסת דאטה יוצרים פיסת דאטה קרובה (נגיד סמנטית בשפה הטבעית או שתי פאצ'ים של אותה תמונה בראייה הממוחשבת). לאחר מכן יוצרים מספר של זוגות של פיסות דאטה שהן לא קשורות אחת לשניה (נגיד פאצ'ים מתמונות שונות). בגדול מאוד המטרה של CI היא לבנות(= לאמן מודל המפיק אותו) ייצוג הדאטה הממזער מרחק בין פיסות דאטה דומות(זוגות חיוביים) וממקסם אותו בין פיסות הדאטה הלא דומות (זוגות שליליים). לפעמים משתמשים במרחק קוסיין, לפעמים מרחק אוקלידי ויש עוד וריאנטים; יש מגוון שיטות לבנות זוגות חיוביים ושליליים אבל העיקרון נותר על כנוץ

אחד המאמרים הראשונים שהשתמשו בלמידה הניגודית בהקשר של בניית ייצוג דאטה הוא [InfoNCE](https://arxiv.org/pdf/1807.03748.pdf) של Oord ושותפיו. לאחר מכן השיטה שימשה מחברים של מאמרים מאוד מפורסמים כמו [SimSCE](https://arxiv.org/abs/2104.08821) ו-[MoCo](https://arxiv.org/abs/2003.04297). עכשיו אתם בטח רוצים לשאול איך CI קשור למאמר שנסקורו היום? כמובן כדי לשפר את האמבדינגס של הטקסט עבור משימות RAG.

ל-RAG לרוב יש שתי בעיות משמעותיות:

1. לא תמיד טקסט בעל ייצוג קרוב (בד״כ לפי מרחק קוסיין) לייצוג השאלה מכיל תשובה על השאלה (או משהו שניתן לגזור ממנו תשובה). ייתכן מאוד שהטקסט שייך לאותו התחום(דומיין) אבל לא מכיל תשובה על השאלה.
2. המרחב של ייצוגי הטקסטים (המופקים על ידי LLM-ים עוצמתיי) הוא מרחב לא טריוויאלי מבחינת המרחק בין הייצוגים ולפעמים מרחקים בין זוגות טקסט קשורים ולא מאוד קשורים עלולים להיות קרובים זה לזה. ז״א כאשר אתם בוחרים פיסות טקסט עם ייצוג הקרובים ביותר לייצוג השאלה לא תמיד מקבלים פיסות טקסט רלוונטיות.

אז הפתרון הגיוני ביותר הוא לכייל (fine-tune) מודל שפה על דאטהסט שאלות ותשובות מדומיין (משימה) כך שיקרב את ייצוגים של שאלות ותשובות רלוונטיות וירחיק את הייצוגים של השאלות והתשובות הלא רלוונטיות. הבעיה הגדולה עם הגישה הזו היא שבנייה ידנית של דאטהסט כזה היא יקרה ולוקחת הרבה זמן.

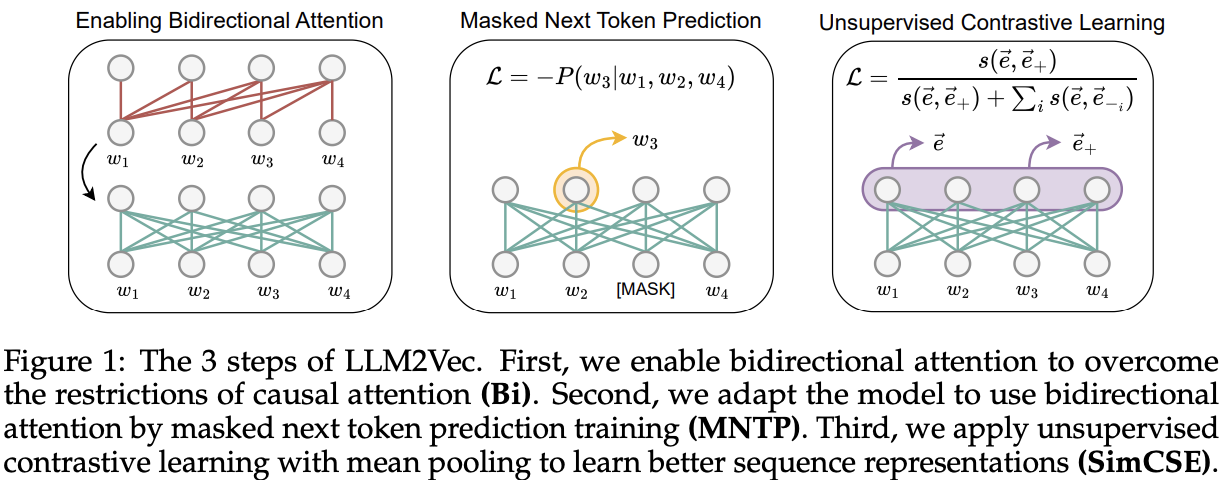
אז למה לא לרתום LLMs למשימה הזו? זה בדיוק מה שעשו המחברים של המאמר המסוקר. הם ביקשו LLM מצוי ליצור זוגות של שאלות עם תשובות נכונות ובנוסף גם זוגות עם תשובות לא נכונות, אבל נראות ״דומה לנכונות״ (hard negatives). היתה שם הנדסת פרומפטים חמודה אבל לא משהו מהפכני במיוחד.

ולבסוף השתמשו בגישת CI סטנדרטית כדי לכייל מודל שפה בדיוק באופן הסברתי לפניכן. לא מצאתי במאמר איך בדיוק מתבצע האימון (הוסיפו שכבות או אימנו כמה מהן) אבל הרעיון ברור.

וזה וזה…

## **Review 199: LLM2Vec: Large Language Models Are Secretly Powerful Text Encoder**

https://arxiv.org/abs/2404.05961



המאמר הזה תפס את תשומת ליבו עקב העובדה שהוא דן בנושא שמאוד מעניין אותי לאחרונה (בנוסף לממבה וחידושים למודלי דיפוזיה🙂). והנושא הזה הוא התאמת מודלי שפה מאומנים לביצוע משימות דיסקרימינטיביות, למשל משימות זיהוי נושא או סנטימנט, זיהוי חלקי דיבור וכדומה. הרי רוב מודלי שפה בתקופה האחרונה מאומנים לגנרט טקסט, כלומר לבצע משימה גנרטיבית(מבוססים על דקודר בלבד).

אתם יכולים להגיד למה צריך מודלים למשימות דיסקרימינטיביות אם ניתן די בקלות להפוך רוב המשימות דיסקרימינטיביות לגנרטיביות? למשל משימת זיהוי סנטימנט ניתן להחליף במשימת גנרטיבית של גנרוט הסנטימנט לטקסט נתון (כלומר ״הסנטימנט בטקסט זה היה חיובי״). אבל נשאלת השאלה האם החלפה זו היא אופטימלית מבחינת הגודל, הביצועים והמאמץ הנדרש לאימון מודל כזה למשימה נתונה. בלא מעט מקרים (למשל כאשר יש דרישות קשיחות לצריכת זיכרון או לייטנסי מקסימלי של המודל).

האם אפשר לעשות יותר טוב? כאמור רוב המודלים החזקים שיצאו ב-3 השנים האחרונות הם מודלים גנרטיביים בעלי ארכיטקטורת הדקודר (gpt, gemini, claude etc). המודלים שאומנו למשימות דיסקרימינטיביות בעלי ארכיטקטורה הכוללת אנקודר הפכו להיות די נדירים לאחרונה. לאור זה המאמר שנסקורו היום מנסה להתאים (לכייל) מודל שפה גנרטיבי (דקודר) למשימות דיסקרימינטיביות.

עכשיו נשאלת השאלה למה לא לקחת מודל שאומן כדקודר וישר לעשות לו פיינטיון (fine-tune) למשימה דיסקרימינטיבית? כדי להבין למה זה עלול להיות לא אופטימלי צריך להרחיב טיפה על איך בדיוק מאמנים מודלי אנקודר ומודלי דקודר.

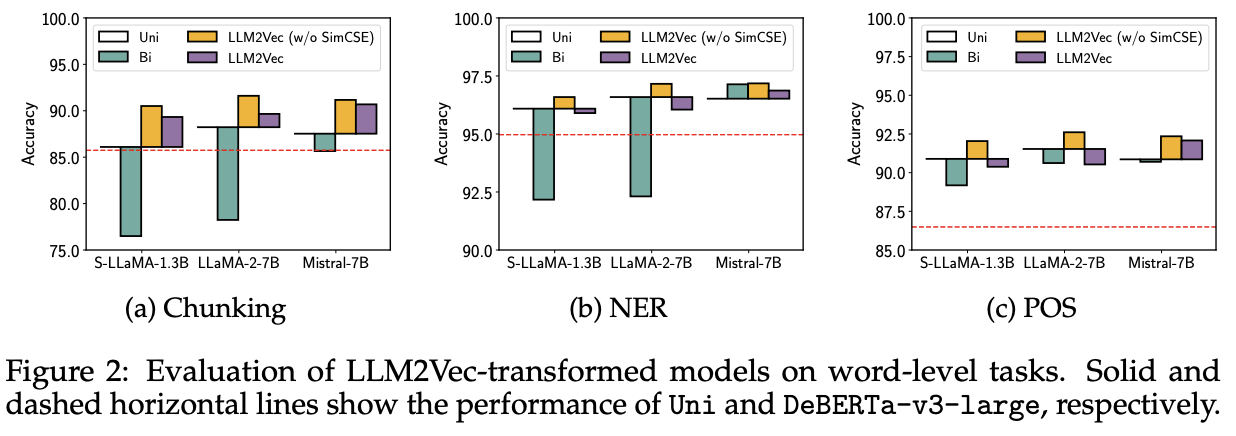
במהלך אימון האנקודר אנו ממסכים טוקנים מסוימים ומאמנים את המודל לחזות אותם. כלומר אנחנו משתמשים **בכל הטוקנים בטקסט** כדי לחזות את הטוקנים הממוסכי. אם הדאטהסט שאנו מאמנים עליו גדול ומגוון מספיק המודל לומד ״להבין״ (לאפיין סטטיסטית) את השפה. לעומת זאת מודל הדקודר הינו מודל גנרטיבי כלומר המודל יוצר פיסות דאטה חדשות. זה מצריך אופן אימון שונה מהאנקודר. הדקודר מאומן לגנרט דאטה חדש: המודל מאומן לחזות את המילה (טוקן) הבא. כלומר להבדיל מאופן אימון האנקודר אנו **מסתירים מהמודל את הטוקנים שבאים אחרי הטוקן הנחזה**, **כלומר חוסמים ממנו את העתיד**.

מכאן ניתן לראות עקב אופן אימון שונה קשה וקצת נאיבי לצפות מהמודלים שמאומנים כדקודרים להצטיין במשימות דיסקרימנטיביות אחרי פיינטיון (אני לא טוען שזה בלתי אפשרי וכנראה יש משימות שזה יעבוד להם לא רע, כמובן זה תלוי בכמה דאטה מתיוג יש). נגיד למשימה זיהוי של חלקי דיבור הייצוג של מילה במודל הדקודר המאומן (pretrained) לוקח בחשבון רק את המילים הקודמות שכמובן לא אופטימלי עבור משימה זו.

אחרי הקדמה ארוכה זו בוא נתמקד במאמר המסוקר. כאמור הוא מציע דרך להתאים מודל דקודר מאומן למשימות דיסקרימנטיביות. המאמר מציע 3 שלבים ל״הפיכה״ של מודל דקודר למודל האנקודר:

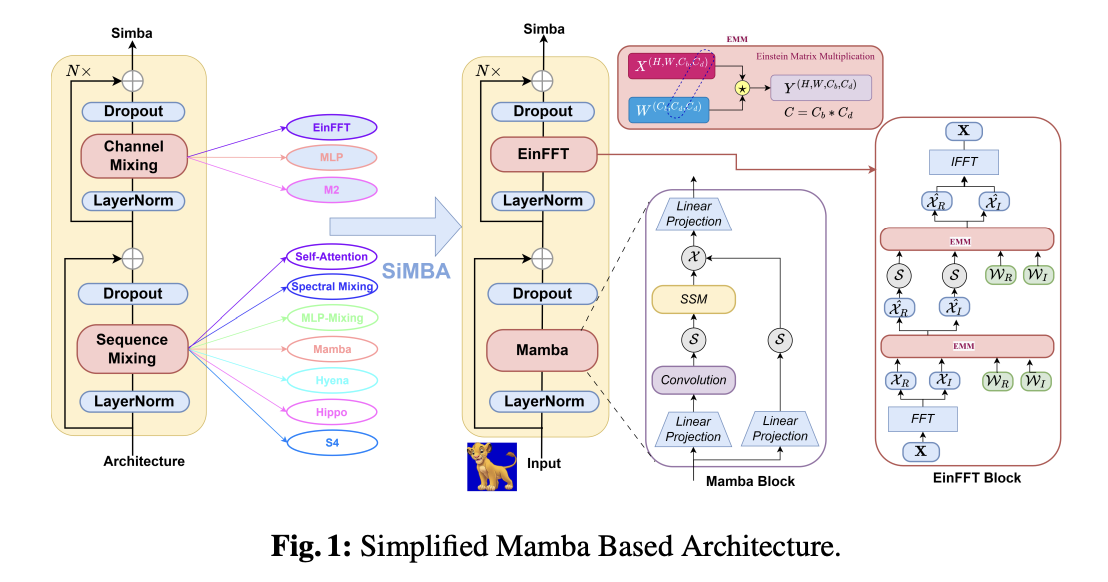
1. ביטול איפוס הטוקנים העתידיים במנגנון ה-attention כלומר המודל חופשי לנצל את כל הטוקנים לבניית ייצוג של כל טוקן . ד״א המאמר טוען הביצועים של המודל לאחר מכן יורדים (בגלל זה יש עוד 2 שלבים בתהליך).
2. במהלך האימון במקום לחזות את הטוקן הממוסך מייצוגו ההקשרי (contextualized) אנו עושים זאת מייצוגו של הטוקן הקודם. לא ברור לי ב 100% מה ההיגיון מאחורי זה.
3. שימוש בלמידה ניגודית (contrastive learning). גישות למידה ניגודית משמשות לאימון של ייצוג דאטה (לא מתויג בד״כ) כאשר מטרת האימון לקרב ייצוגים של פיסות דאטה קרובות ולהרחיק ייצוגים של פיסות דאטה לא דומות/לא קשורות (מבחינת דמיון קוסיין). אז המאמר מציע לאמן את המודל לקרב ייצוגים של אותו המשפט עם drop-outs שונים (בגדול מאוד dropout הוא למעשה איפוס קשרים/משקלים בין נוירונים שונים במודל. לעומת זאת ייצוגים של משפטים מאומנים להיות רחוקים אחד מהם במרחב אמבדינג.ֿ

לטענת שילוב שלבים אלו הופך את המודל שלכם לאנדוקר המסוגל להפיק ייצוגים דאטה חזקים המפגינים ביצועים לא רעים בכמה משימות דיסקרימינטיביות.



## **Review 200: SiMBA: Simplified Mamba-based Architecture for Vision and Multivariate Time series**

https://arxiv.org/abs/2403.15360



המאמר הזה משך את תשומת ליבי כי מצד אחד יש בו שימוש נרחב בהתמרת פורייה ובייצוגים של דאטה בתחום התדר. החולשה שלי לתחום התדר נובעת מכך שביליתי כמה מהשנים הראשונות של הקריירה בתחום עיבוד של אותות אלחוטיות. מצד שני המאמר גם משתמש בארכיטקטורת ממבה שסקרתי בהרחבה בחודשים האחרונים (וכנראה אמשיך עם זה כי מאמרים מעניינים בנושא מרתק זה לא מפסיקים להגיע).

אוקיי, אז מה יש לנו במאמר הזה? המאמר מציע שדרוג יפה לארכיטקטורה של ממבה המערב כאמור התמרות פורייה וקצת משחקים בתחום התדר. הארכיטקטורה המוצעת מתאימה גם לדאטה ויזואלי וגם לסדרות זמן multivariate. המאמר כתוב בצורה די מסורבלת והיה לי לא טריוויאלי לגלות מה הם באמת עשו עקב הסברים וסימונים לא ברורים. אבל כאמור הרעיון מאחורי המאמר הוא די חמוד.

המחברים מנסים לשפר את ממבה על ידי הוספת שכבה שבגדול לוקחת את הייצוגים המופקים על ידי ממבה ו״מחזקת״ אותם על ידי פלטור תדרים מסוימים מהם (הייצוגים). קודם כל נציין שמפעילים את המנגנון המוצע, שקיבל שם EinFFT, על כל איבר סדרה בנפרד (פאץ' של תמונה) בצורה ממוקבלת. כאמור הסיפור מתחיל מהפעלת התמרת פורייה על הפלט(=ייצוג פיסת דאטה) של שכבת ממבה. ואז המאמר הופך להיות די לא ברור והדבר הזה גזל ממני בערך שעותיים כדי להבין שלא אני מפספס משהו אלא המאמר עצמו קצת לא מדייק (בתקווה עמדתי במשימה זו).

כאמור הרעיון הוא מפלטר תדרים(תלויות) הלא נחוצים (לביצוע המשימה) בייצוגי איברי הסדרה. הפלטור מתבצע במחרב הייצוג של הדאטה (כלומר אמבדינג) ונקרא channel-mixing. כלומר שכבה זו היא משמשת בתור תוספת/החלפה ל-MLP שלפעמים משמשת לאותה המטרה.

אבל איך הוא עושה את זה משתנה בין נוסחה לנוסחה במאמר. במאמר עצמו (נוסחה 4) קודם כל מפעילים שכבה לינארית במישור המרוכב ולאחריה סיגמויד (גם במישור המרוכב). ב-appendix (בתחילת עמוד 22) זה כבר מופיעה שכבה אחת של ReLU, לאחר מכן עוד שכבה לינארית, לאחר מכן מפעילים פונקצית [softshrink](https://pytorch.org/docs/stable/generated/torch.nn.Softshrink.html) שמטרתה היא לאפס תדרים סביב אפס באינטרבל באורך 2\*lambda ולהזיז כאלו מעבר לזה ב-lambda. כלומר איזה stop-band filter מוזז.

הגרסה השלישית מגיע מהדף האחרון של ה-appendix ששם יש רק ReLUs. לא הסתכלתי בקוד אז לא ברור מה באמת קורה שם. כל הפעולות האלו מתבצעות בצורה נפרדת במישור הממשי ובמישור המדומה שלאחר מכן משלבים אותם. בשלב האחרון מבצעים התמרת פורייה הפוכה (IFFT).

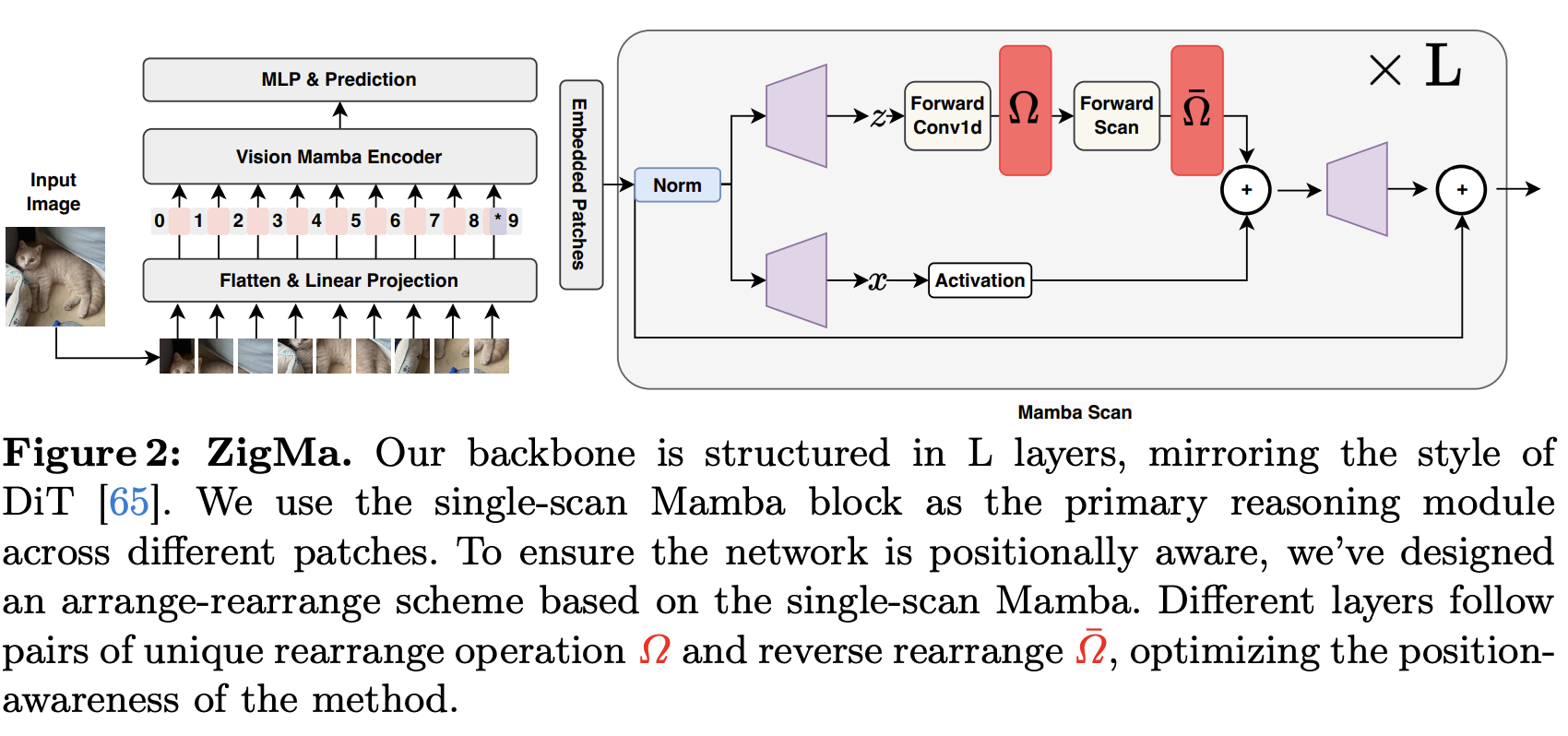
אוקיי, אז בואו נחזור לעיקר. המנגנון שבא אחרי שכבת ממבה נקרא EinFFT וכבר הבנו ש- FFT מתאים להתמרת פוריה. אבל מה זה Ein? באופן לא מפתיע אלו 3 האותיות הראשונות נלקחו מאינשטיין. אז מה בעצם איינשטיין עושה כאן?

למעשה המאמר משתמש בסכימת איינשטיין שהיא דרך לרשום מכפלות הטנזורים או המטריצות במקרה פרטי. למעשה במקום לרשום כל איבר ij של המכפלת מטריצות A ו- B בתור מכפלה פנימית של שורה i ועמודה j סכימת איינשטיין כותבת אותו ללא סימן של סכום(=סיגמה) אלא על ידי ציון של מספר שורה i, מספר עמודה j ואינדקס סכימה k.

אז איך המאמר משתמש בסכימה הזו? הרי אמרתי שהסכימה הזו מוגדרת גם לטנזורים ומתברר שלחבילות תוכנה כמו pytorch יש חבילות שיודעות לבצע מכפלת טנזורים רב מימדיים המבוטאים דרך סכימת איינשטיין בצורה די יעילה. וזה בדיוק מה שעושים במאמר. המאמר מפרק את המטריצות מהשכבות הלינאריות של EinFFT לכמה מטריצות במימד נמוך יותר ובונה מזה טנזור רב מימדי הבנוי ממטריצות בלוקיות (אפסים מחוץ לבלוקים). הטענה במאמר שזה מאפשר לבצע את המכפלות (באימון אבל כנראה גם באינפרנס) בצורה מהירה יותר על ידי ניצול טוב יותר של משאבי החומרה.

הביצועים כרגיל מפתיעים לטובה….

## Review 201: ZigMa: A DiT-style Zigzag Mamba Diffusion Model https://arxiv.org/abs/2403.13802



המאמר הזה משך את תשומת ליבי מכמה סיבות:

1. יש מודלי דיפוזיה - האהבה הקודמת שלי שבקרוב מאוד אחדש את הקשר איתם
2. יש כאן (SSMs (State-Space Models בדמות Mamba - האהבה הנוכחית שלי שתיכף אני מסיים להכין עליה מצגת די רצינית ובתקווה ישמע אותי מציג אותה בפורומים השונים
3. המאמר פורסם בראשון לאפריל ובהתחלה קצת חשדתי 🙂

בנוסף יש במאמר גם קצת מהטרנספורמרים (cross-attention) שעוד מוסיף לשלמותו. אוקיי, אז מה יש לנו במאמר הזה מעבר לכמה מילים ״באזזיות״. המאמר מציע ארכיטקטורה מעניינת המיועדת לגינרוט תמונות וגם וידאו. כאמור הארכיטקטורה היא שייכת למשפחה של מודלי דיפוזיה גנרטיביים אבל מכילה חלקים המורכבים מ-SSMs (ממבה) בנוסף ל-cross-attention הלב של הטרנספורמרים. ויש כאן חידוש מעניין לגבי הסדר שבו מכניסים פאצ'ים של תמונות (או פריימים של וידאו) במהלך אימון המודל.

נתחיל מהסבר קצר על מודלי דיפוזיה גנרטיביים. בהינתן דאטהסט (של תמונות או/ו סרטונים) אנו מאמנים את רשת באופן הבא:

1. מוסיפים כמויות קטנות של רעש גאוסי לפיסת דאטה עד שהיא היא הופכת לרעש טהור
2. מאמנים רשת נוירונים (עם Mamba ו-cross-attention במקרה שלנו) כדי למדל של התהליך ההפוך: כלומר מפיסת דאטה מורעשת מאיטרציה n לחזות אותה באיטרציה n-1.

כאשר יש בידינו מודל כזה אנו למעשה מסוגלים לגנרט תמונה מרעש גאוסי טהור בצורה הדרגתית, איטרציה אחרי איטרציה. עם השנים צצו שיטות רבות ומגוונות מאוד לאיך להוסיף רעש ומה בדיוק כדאי לחזות עם הרשת שלנו.

בשנה וחצי האחרונות היו כמה חידושים מעניינים במודלי דיפוזיה ומכיוון שהמאמר משתמש בהם אני חייב לספר לכם בגדול במה מדובר (כאמור הולך לדבר על זה בהרחבה בסקירות הבאות).

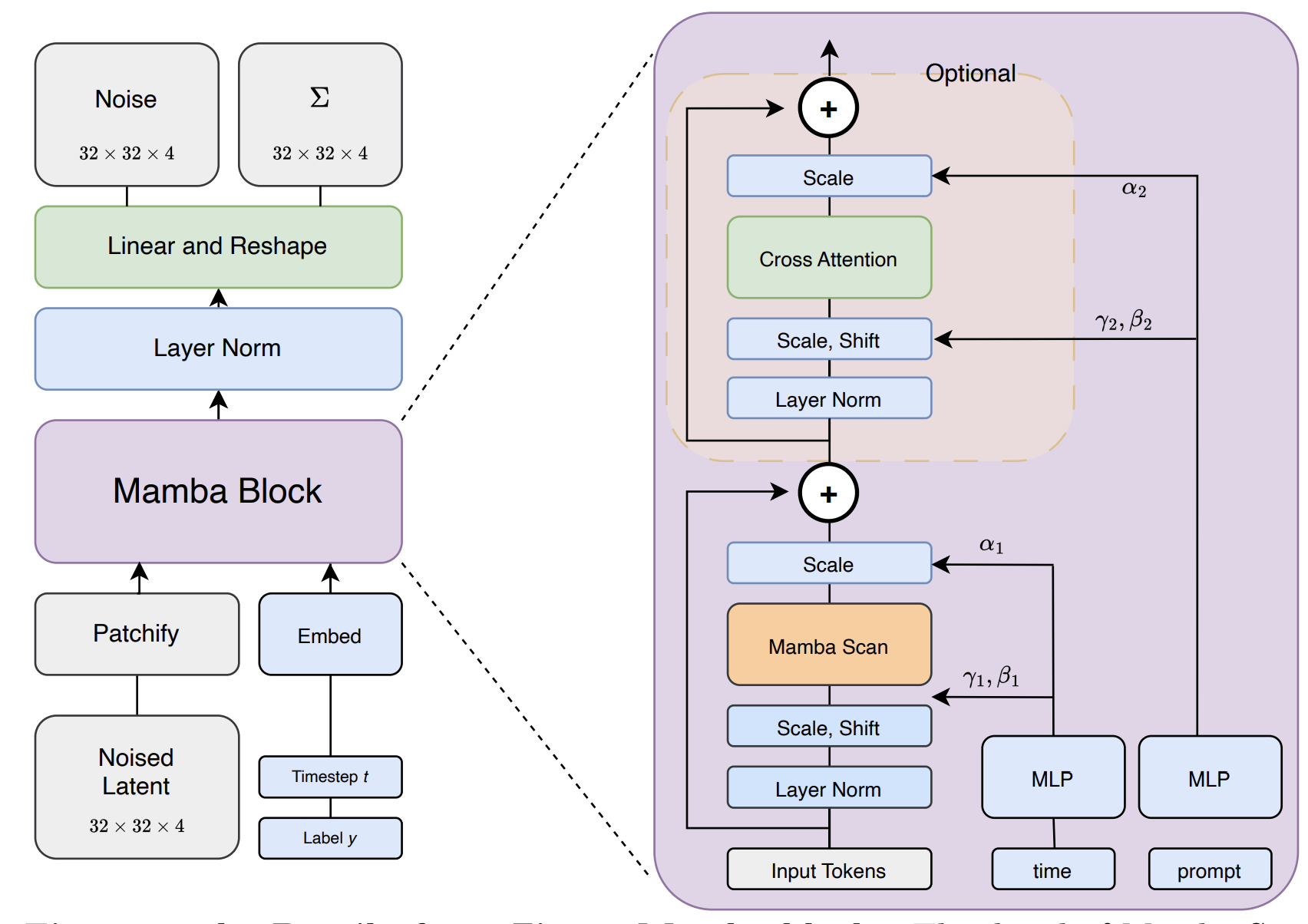
לאחרונה יצאו כמה מאמרים מעניינים (למשל <https://arxiv.org/abs/2210.02747> ו- <https://arxiv.org/abs/2303.08797> אבל יש עוד עשרות אחרים) המכליל מודלי דיפוזיה לתהליך רציף של מיפוי התפלגות פשוטה (כגון גאוסית סטנדרטית) להתפלגות של הדאטה (ההתפלגות המורכבת). תהליך זה נקרא זרימה רציפה (continuous flow) הדיסקרטיזציה שלו (במישור הזמן כלומר האיטרציות) היא מודל דיפוזיה גנרטיבי עבור מיפוים מסוימים. יש לנו חופש לבחור את המיפוי (זרימה) בין התפלגות דאטה להתפלגות הפשוטה ויש לא מעט מחקרים על איך לבחור אותו בצורה אופטימלית (למקסם את איכות הדאטה המגונרטת, לייצב את התהליך, ליצור מיפוי כמה שיותר פשוט או ישר וכדומה).

אז איך כל המתמטיקה הזו קשורה לגנרוט דאטה? אז יש כאן עוד קצת מתמטיקה שנצטרך לצלול בה. בגדול הזרימה הרציפה בין להתפלגות הפשוטה להתפלגויות הדאטה (לפעמים נקראת reverse-time) ניתן לתאר על ידי משוואה דיפרנציאלית סטוכסטיות (SDE) שמכילה:

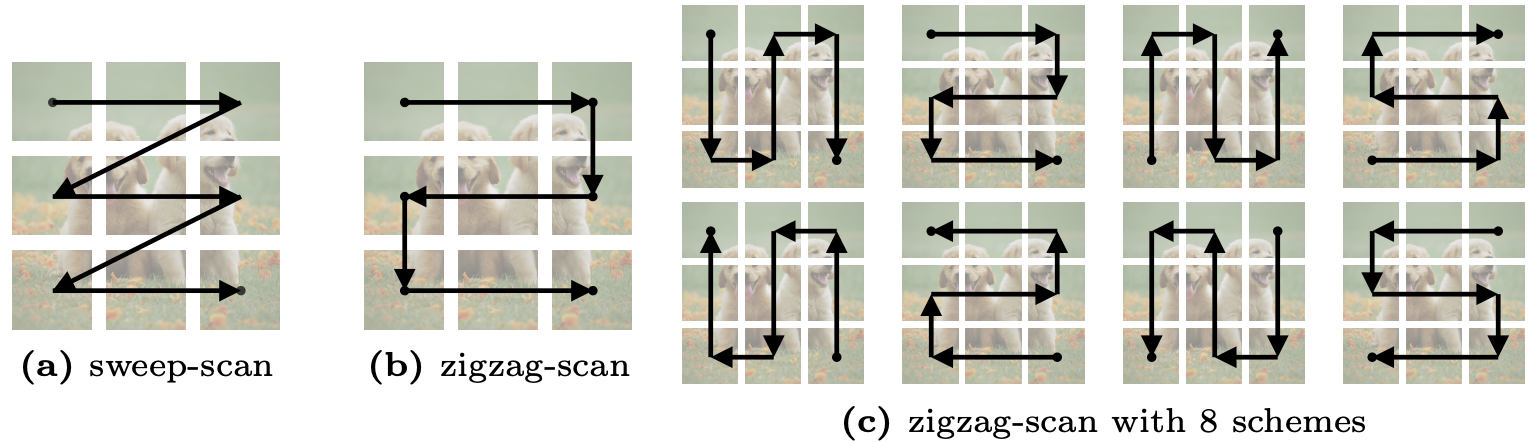
1. הדאטה המורעש עצמו x\_t
2. מהירות או השתנות(נגזרת בזמן) של הזרימה בזמן (v\_t(x (תחשבו על זה כמו על תנועה במרחב בין 2 עננים של נקודות שניתן להגדיר אותה על ידי המהירות הכיוונית ונקודה ההתחלה או הסוף).
3. פונקציית (score (s\_t(x שהיא בעצם לוגריתם של (p\_t(x - פונקציית התפלגות של הדאטה המורעש
4. יש גם תהליך reverse-time Wiener המהווה את החלק אקראי (סטוכסטי) ב-SDE הזה

אז מה אפשר לעשות עם ה-SDE הזה, למה צריך אותו? מתברר כי עבור פרמטרים של הזרימה בין התפלגות הדאטה להתפלגות הפשוטה ניתן לנסח בעיות אופטימיזציה המאפשרות שערוך של (s\_t(x ו- (v\_t(x בהינתן דאטה לאימון. אחרי שנשערך אותן ניתן לפתור את ה-SDE שדיברנו עליה נומרית (נגיד בשיטת אוילר-מרואימה) כלומר מנקודת התחלה הנדגמת מההתפלגות הפשוטה (גאוסית) נוכל לגנרט דאטה צעד אחרי צעד. וזה בדיוק מה שעושים במאמר.

אוקיי, שרדנו את המתמטיקה - עכשיו מה הקשר ל-SSMs כאן? בשביל כך צריך להיזכר בארכיטקטורה של Diffusion Transformer או DiT, זה שעליו מבוסס המנוע של SoRA של OpenAI. למעשה DiT מורכב מהבלוקים של טרנספורמרים שמטרתם היא למדל את הפרמטרים ( (s\_t(x ו- (v\_t(x) (כמובן לאחר דיסקרטיזציה במישור הזמן, כלומר איטרציות). המאמר המסוקר מחליף את בלוקי הטרנספורמר ב-Mamba (בנוסף הם גם לוקחים cross-attention שזה הלב של הטרנספורמר אך לפי הציור שלהם החלק הזה הוא אופציאונלי).



אבל כאן יש לנו בעיה. מכיוון שממבה היא ארכיטקטורה מיועדת לסדרות בעלת מימד הזמן חד מימדי (למשל טוקנים של טקסט) כאן יש לנו תמונות ובהן קיימים קשרים דו מימדיים בין הפאצ'ים (טוקנים ויזואליים) בתמונה וקשרים תלת מימדיים בוידאו (בנוסף בין הפריימים). המאמר מתאים את המבנה של SSM עבור הקלט בעל קשרים רב מימדיים על שילוב של SSM-s שכל אחד מקבל את הקלט בסדר שונה (תראו בתמונה). כלומר שכבות של ממבה מוערמות (stacked) אחת מעל השנייה כל הקלט נכנס לכל אחד מהם בסדר שונה (למיטב הבנתי כל הממבות עובדות עם אותם מטריצות הפרמטרים A, B, C). זה מאפשר ל- ZigMa להתחשב בקשרים האלו. המאמר מרחיב את הגישה הזו לגנרוט וידאו (עבור קשרים תלת-מימדיים).



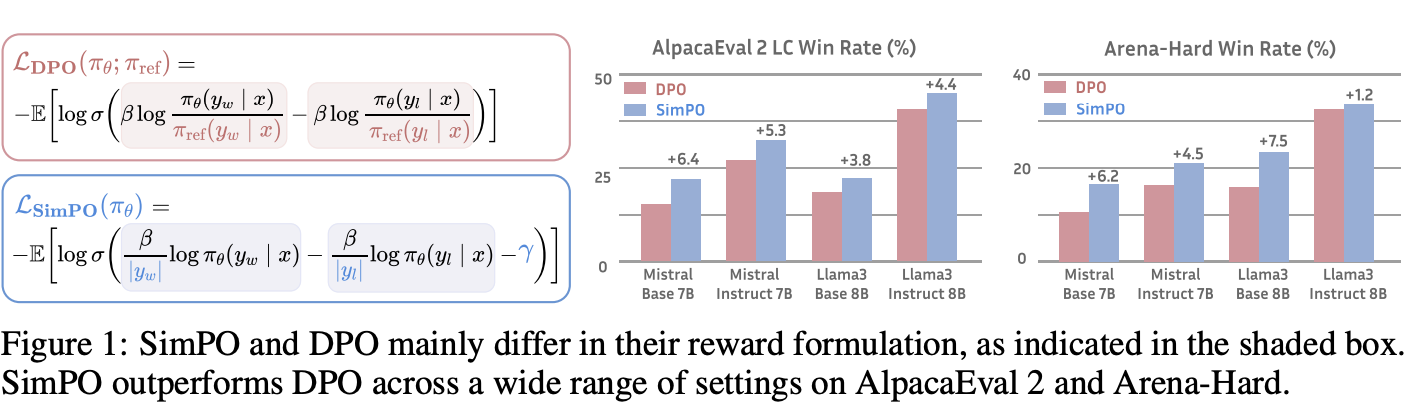
אציין שבדומה ל-DiT המודל המוצע פועל במרחב הלטנטי כלומר הקלט למודל דיפוזיה הוא ייצוג לטנטי של הדאטה אחרי האנקודר. DiT משתמש באנקודר ובדקורר של VAE (אחד השכלולים שלו) אך במאמר הזה לא הצלחתי להבין האם המחברים לקחו VAE. במקום אחד במאמר רומזים שהאנקודר גם מכיל SSM אבל לא מצאתי לזה אזכורים נוספים.

התוצאות נראות לא רע, לשנות 2020 ככה אבל מכיוון שזה אחד המאמרים הראשונים המשלבים SSMs ומודלי דיפוזיה נסלח להם על כך.

יצאה סקירה ארוכה אבל מובנת פחות או יותר בתקווה…

## **Review 202: SimPO: Simple Preference Optimization with a Reference-Free Reward**

https://arxiv.org/abs/2405.14734



המאמר שנסקור דן בנושא אימון של מודלי שפה. אתם בטח יודעים שאימון מודל שפה foundational מורכב מ- 3 שלבים עיקריים:

1. אימון מודל self-supervised על דאטהסט ענק
2. אימון(פיינטיון) מפוקח (supervised fine-tuning או SFT) על דאטהסט מתויג קטן יותר (בד״כ מכיל תשובות רצויות למגוון שאלות) במטרה לגרום למודל לעקוב אחרי הוראות המשתמש (instruction following)
3. שלב RLHF: מתברר שרוב המודלים לא מצליחים ללמוד רק מהתשובות ה״טובות״ ואנו נדרשים לספק לו גם את התשובות ה״לא טובות״. השלב האחרון נעשה באמצעות שימוש בטכניקות השונות של למידה עם חיזוקים.

המודלים הראשוניים (גוגל, OpenAI) שהשתמשו ב-RLHF ליישור (alignment) של המודלים התבססו על טכניקה שנקראת Proximal Policy Approximation או PPO בקצרה. במהלך האימון אנו מעדכנים את המודל שלנו כך שהוא ייתן תגמול (=reward) גבוה לתשובה טובה לתשובה ותגמול נמוך לתשובה לא טובה תוך שמירה של המודל החדש קרוב (מבחינת התפלגויות הטוקנים) שהוא מוציא להתפלגות המתקבלת בשלב 2.

אבל איך נמדוד את התגמול הזה? עבור PPO אנו צריכים לאמן מודל תגמול שבהינתן פרומפט ותשובה יחזיר לנו ציון (סקלרי). עבור תשובה טובה הציון יהיה גבוה ועבור תשובה לא טובה הוא יהיה נמוך. מאמנים את המודל הזה על הדאטהסט של התשובות הטובות ולא טובות משלב 3.

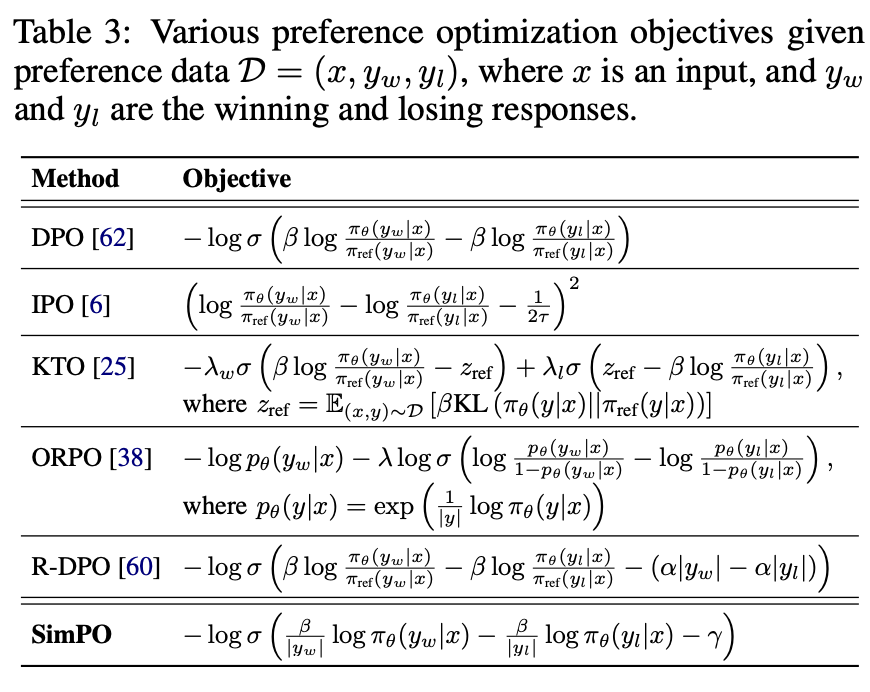
כמובן שאם היה אפשר להסתדר ללא מודל תגמול מצבנו היה טוב יותר. קודם כל זה חוסך לנו את זמן ומשאבים ובנוסף אנו לא צריכים להפעיל אותו לאינפרנס במהלך אימון RLHF שזה גם יכול להפחית את דרישות הזכרון וכוח חישוב. אז הוצעו שיטות כמו Direct Preference Optimization או DPO שהוריד את הצורך באימון מודל תגמול. לאחר מכן יצא מודל הנקרא ORPO (סקרנו אותו באנגלית לפני כחודש) הסתדר גם בלי להשתמש במודל משלב 2 במהלך האימון (משמש רק לאתחול המודל משלב 3).

עכשיו הגענו למאמר המסוקר. הוא הציע שכלול ל-DPO הנקרא SimPo. כמו OrPo הוא לא צריך מודל רפרנס בצורה מפורשת במהלך אימון שלב 3 ומציע לאמן את המודל על ידי מקסום ההפרש בין התגמול של התשובה הטובה והתשובה הלא טובה (עם הסיגמויד) עם איזשהו מרג'ין מסוים. החידוש העיקרי של המאמר שבתור פונקציית תגמול המחברים לוקחים את הנראות המירבית של תשובה בהינתן שאלה, **המנורמלת באורך התשובה** (בטוקנים). המחברים טוענים שדבר זה (נרמול) בין השאר מונע מהמודל לגנרט תשובות ארוכות מדי וזה אכן נשמע די הגיוני.

דרך אגב בנוגע להמרג'ין נטען המאמר שמספר עבודות קודמות ציינו שזה מיטיב עם תהליך האימון (למרות שזה די הוספת קבוע).

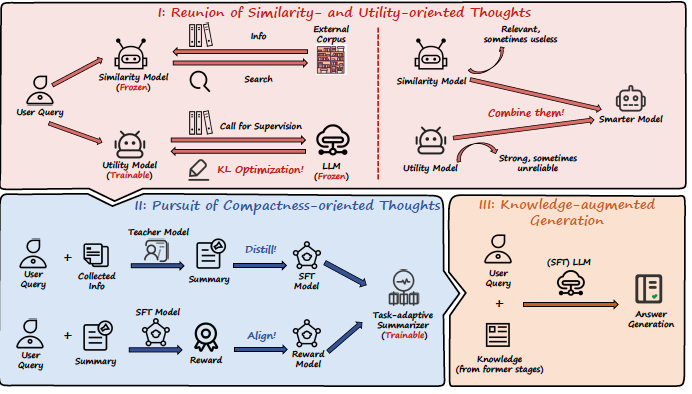
*״The margin between two classes is known to influence the generalization capabilities of classifiers [1, 9, 19, 27]. In standard training settings with random model initialization, increasing the target margin typically improves generalization״.*

יש גם את הטבלה החמודה הזו המסכמת את רוב המחקרים האחרונים בתחום RLHF למודלי שפה.



## **Review 203: Similarity is Not All You Need: Endowing Retrieval-Augmented Generation with Multi–layered Thoughts**

<https://arxiv.org/abs/2405.19893>



בזמן האחרון גישות המשלבות מודלי שפה עם בסיסי נתונים חיצוניים הפכו למאוד פופולריים. גישות אלו לרוב שייכות למשפחת Retrieval Augmented Generation או RAG בקצרה. בגדול בהינתן מודל שפה ומסמכים העשויים להכיל תשובה על שאלת משתמש, RAG קודם מחפש כמה מסמכים הרלוונטיים ביותר לשאלה ואז מזינה אותם יחד עם השאלה למודל שפה. המודל מרכיב את תשובתו על השאלה בהתבסס על המסמכים שהוזנו אליו.

אבל איך נבחר מסמכים הרלוונטים יותר לשאלה? בדרך כלל בוחרים אותם לפי הקרבה של האמבדינג (= ייצוג וקטורי) שלו לאמבדינג של השאלה. בדרך כלל המציאות טיפה יותר מורכבת ממה שתיארתי: למשל אם המסמכים ארוכים צריך לחלק אותם לצ'אנקים אז הבחירה היא לפי דמיון האמבדינגס של הצ'אנקים לזה של השאלה. כמובן שיש עוד גישות.

הדמיון בין אמבדינגס בד"כ מחושב לפי דמיון קוסיין (זווית בין הוקטורים). האם הבחירה הזו היא אופטימלית - זו השאלה שהמאמר שנסקור היום מנסה לענות עליה.

כדי להבין האם הבחירה אופטימלית צריך להגדיר מדד אופטימליות. הרי בסופו של דבר מטרתנו היא לתת תשובה נכונה לשאלת המשתמש. המאמר טוען שבחירת מסמכים רלוונטים לפי דמיון אמבדינגס אינו אופטימלי בהתאם המדד הזה. אז המחברים מציעים גישה לשכלול הבחירה של המסמכים הרלוונטים לשאלה.

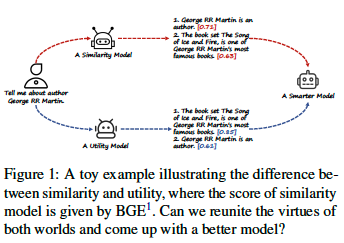
האמת הם מציעים משהו די טבעי - בגדול המטרה שלהם היא לאפטם את הביצועים של RAG דרך ״מקסום הסיכוי לקבלת תשובה טובה אחרי בחירת מסמכים רלוונטים על ידי RAG״. המחברים מנסים להשיג את המטרה בכה שלבים:

**שלב 1: אימון מודל utility**. המטרה של מודל זה להעניק ציון ליכולת של מסמך נתון "לתת' תשובה טובה לשאלה כאשר הם (המסמך והשאלה) מוזנים למודל שפה יחד. אבל איך נדע לשערך את איכות התשובה? בשביל זה המחברים לקחו מודל שפה חזק (נגיד gpt4) שמטרתו היא לתת ציון לתשובה עבור מסמך ושאלה נתונים (ככל שהתשובה טובה ציון גבוה יותר). המאמר לא מסביר איך זה נעשה אבל אני מניח שעבור דאטהסט המכיל תשובות ניתן למדוד דמיון סמנטי בין תשובה אמיתית לתשובה מופקת על ידי llm (כלומר בין האמבדינגס), ניתן גם למדוד אותה על ידי הזנתם של המסמך, השאלה והתשובה ל-llm ומדידת נראות מירבית שלה (כלומר logits), בטח יש עוד שיטות. המחברים מאמנים utility model (שהוא מודל קל יחסית) להחזיר את אותה ההתפלגות של ציוני מסמכים (בהינתן שאלה) כמו המודל החזק. כלומר ממזערים KL divergence בין התפלגות ציונים של utility model לבין זו של מודל השפה (שהוא מוקפא - לא מאומן).

**שלב 2: בחירת מסמכים** עבור שאלה נתונה בוחרים רק מסמכים שיש להם ציון דמיון או ציון של utility model גבוה מספיק (בין k הגבוהים ביותר כל אחד).

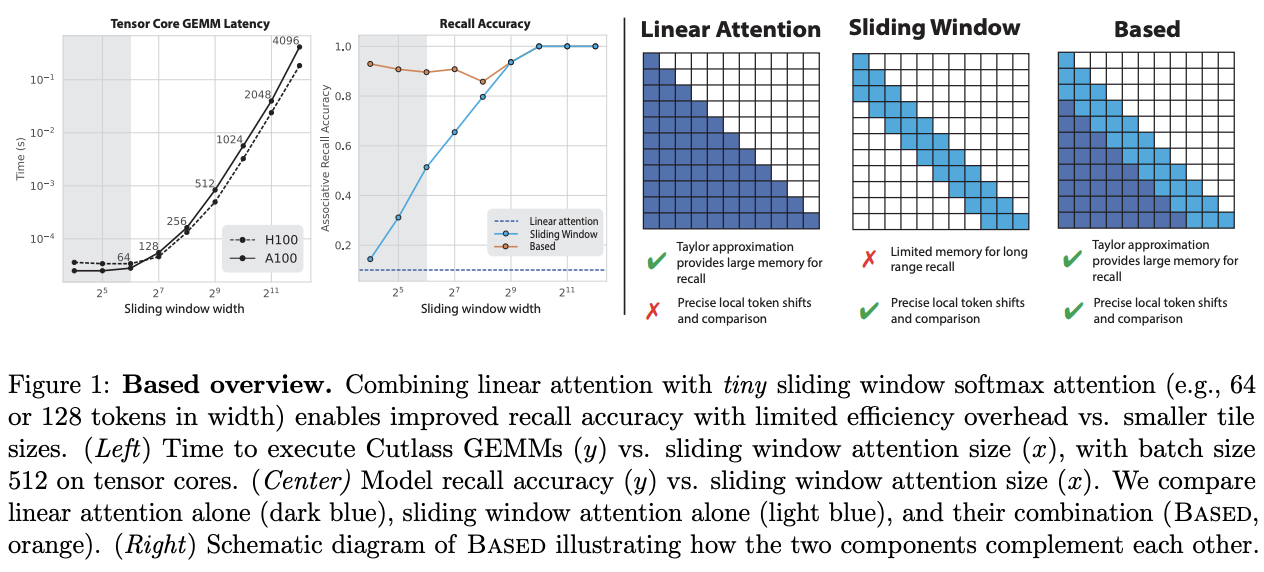
**שלב 3: אימון מודל תמצות מסמכים.** המחברים טוענים שבד״כ המסמכים שנבחרים מכילים לא מעט מידע לא רלוונטי לשאלה שמקשה על מודל שפה לתת תשובה טובה וגם מעלה עליות (צריכים להכניס הרבה טוקנים ל- LLM). במטרה להתמודד עם הקושי הזה המחברים מציעים לאמן מודל שבהינתן שאלה מפיק מהמסמכים שנבחרו את המידע הרלוונטי לשאלה. זה נעשה ב 2 שלבים: בשלב הראשון עבור דאטהסט של שאלות והמסמכים הרלוונטיים מתשאלים מודל שפה חזק (gpt4) לתמצת את המסמכים האלו (עבור שאלה נתונה). על הדאטהסט הזה (שאלה, מסמכים ותמצית) עושים פיינטיון של מודל שפה לא כבד עם LoRa כמובן - כלומר עושים Supervised Fine-Tuning או SFT. בשלב השני עושים RLHF עם DPO כמו שמקובל היום 🙂. בשביל באמצעות מודל שפה(הם לא מפרטים יותר מדי כאן) בונים דאטהסט של תשובות נכונות ולא נכונות בהינתן שאלה ותמצית מסמכים. בניית פונקציית תגמול (reward) מתבצעת בדיוק כמו ב- DPO הסטנדרטי.

אחרי שסיימנו לאמן את מודל התמצות, ההיסק (אינפרנס) נעשה בצורה מאוד טבעית. לוקחים שאלה, מפיקים את המסמכים הרלוונטיים משלב 1, מתמצתים אותם עם המודל משלב 3 ואז מזינים אותם לעוד מודל שפה (המחברים לא מפרטים עליו אבל מציינים שניתן לכייל אותו על דאטהסט כלשהו של שאלות ותשובות). והמודל מספק לנו את התשובה…



## **Review 204: Simple linear attention language models balance the recall-throughput tradeoff**

https://arxiv.org/abs/2402.18668



מודלי שפה ענקיים של היום מפגינים יכולת מרשימה של למידת in-context כלומר יכולת לבצע משימות חדשות (שלא אומן עליהם באופן מפורש) בהתבסס על כמה דוגמאות המדגימות (ממחישות) את את המשימה. כמובן דוגמאות אלו מוזנות למודל שפה כפרומפט. המאמר שנסקור היום מדבר על משימת in-context ספציפית הנקראת recall. המטרה של משימה זו היא לזהות חוקיות מסומיות בפרומפט ולענות על שאלות בנוגע אליו. למשל אם פרומפט המוזן הוא "A 4 B 3 C 6 F 1 G 2". אם לאחר מכן אני מכניסים למודל שפה "?B ? F ? C " המודל צריך לענות 3 1 6 כלומר המספר בא מיד אחרי כל אות בפורמפט השאלה.

ארכיטקטורת הטרנספומרים מתמודדת בהצלחה עם משימות recall אך היא מתקשה עם אורכי הקשר (context length) מאוד ארוכים עקב מנגנון self-attention שלהם. ד״א המימושים המודרניים של מנגנון זה (כמו FlashAttention2 ו-Paged-Attention) הם בעלי סיבוכיות subquadratic במונחי אורך הסדרה אך עדיין גם הם מתקשים ״לעכל״ אורכי הקשר ממש ארוכים.

כדי לתת מענה לסוגיה זו הוצעו מספר חלופות למנגנון ה-attention כמו attention לינארי, שיטות המבוססות על חלון הזז (sliding window) ובנוסף לאחרונה משפחת ארכיטקטורות ממבה (סקרתי אותן בהרחבה לפני כחודשיים).

מנגנון attention לינארי בגדול מחליף את הסופטמקס של המכפלה הפנימית של וקטורי שאילתה (Q) ווקטורי ערך (K) למכפלה הפנימית של (f(Q ו- (f(K עבור פונקציה לא לינארית f (יש לא מעט מאמרים המציעים לקחת פונקציות f שונות עבור ההחלפה הזו). אחת הדוגמאות היא לבחור f בתור כמה איברים ראשונים של פיתוח טיילור של סופטמקס.

פעולה זו מאפשרת להחליף סדר הפעולות בחישוב ה-attention ולבצע את החישוב באופן לינארי במונחי אורך הסדרה. דרך אגב החלפה זו היא כמו reparameterization trick ב- SVMs אבל בכיוון ההפוך. היא מאפשרת להיפטר מ״גרירה״ של הייצוגים של כל הטוקנים הקודמים באופן מפורש באינפרמס ומאפשרת חישוב בסגנון RNN. כלומר כל הזכרון עד טוקן n נדחס לכדי 2 וקטורים (ממליץ לקרוא על זה [כאן](https://arxiv.org/abs/2006.16236)) וכמובן זה מאפשר לחסוך במשאבי חישוב הנדרשים לביצוע אינפרנס באופן משמעותי.

מנגנון ה-attention עם החלון הזז הוא פשוט הגבלת גודל ההקשר במנגנון ה-attention כאשר יש מגוון גישות ל"איך לדחוס״ את הדאטה שלא נכנסת לחלון זה (העבר). בתוך החלון ה-attention מחושב באופן רגיל כלומר הגדלה משמעותית של חלון זה משפרת את הביצועים אבל גם כרוכה בביצוע של יותר חישובים.

מצד אחד ארכיטקטורות המבוססת על attention לינארי יודעות להסתדר לא רע עם אורכי הקשר ארוכים מאוד במשימות מסוימות אבל מתקשות לספק ביצועים גבוהים לשאלות בסגנון recall. מצד שני ארכיטקטורות הממשות חלון attention זז מסתדרות יפה עם משימות recall בתוך החלון הזה אולם כדי להביא ביצועים גבוהים עם הקשר ארוך צריך להגדיל את גודל החלון שכאמור כרוך בהקצאה של יותר משאבים ואו/גם ב-latencies גבוהים יותר באינפרנס.

אוקיי דיברנו הרבה על הרקע למאמר אז הגיע הזמן לדבר על מאמר עצמו. קודם כל החמברים מוכיחים באופן תיאורטי (את הקטע הזה הכי אהבתי כאן) כי ככל שאורך הקלט למשימת recall ״המודל צריך לזכור״ (O(N "מידע" כאשר N הוא ״אורך״ של פרומפט ה-recall (זה גם נבדק אמפירית). כלומר זה תקף לכל ארכיטקטורה והשאלה היחידה איך כל מודל (למשל טרנספורמר לינארי, hyena, mamba, s3 ועוד) בונים ומנהלים את הזכרון הזה ואיך הוא משפיע על ביצועי אינפרנס.

לגבי החידוש שהמאמר מציע: המחברים שילבו את ה״טוב״ שיש במנגנון ה-attention הלינארי ובגישת החלון הזז והציעו מנגנון attention חדש הנקרא Based. הם לקחו מנגנון ה-attention הלינארי החסכוני והיעיל מבחינת ניהול הזכרון והוסיפו לו חלון זז קצר יחסית המממש מנגנון attention רגיל של הטרנספורמים. וזה עבד להם לא רע בכלל במשימות recall שונות המצריכות חלון הקשר גדול. בנוסף גם הציעו מספר שכלולים לשיטה זו המאפשרים להריץ אותה בצורה מאוד יעילה על GPUs (למשל בחירת גודל החלון כדי שיהיה ניתן לבצע את החשובים עבור על ידי שימוש רק הזכרון המהיר של GPU.

בסך הכל מאמר די נחמד…



## **Review 205: PanGu-π: Enhancing Language Model Architectures via Nonlinearity Compensation**

<https://arxiv.org/abs/2312.17276>

היום סוקרים מאמר המציע שדרוג לארכיטקטורת הטרנספורמר. כמו שאתם בטח יודעים בלוק של טרנספורמר מורכב משני החלקים העיקריים (פרט לשכבות נרמול):

* מנגנון תשומת לב עצמי בעל ראשים מרובים (multi-head self-attention or MSA)
* שכבת (MLP (fully connected המורכבת משכבה לינארית עם פונקציה אקטיבציה לא לינארית ולאחר מהן שכבה לינארית נוספת (ללא פונקציית אקטיבציה)

כמו שאתם זוכרים מטרת בלוק הטרנספורמר היא להפיק ייצוגים תלויי הקשר של טוקני הקלט. כלומר כל ייצוג של כל טוקן לוקח בחשבון את הטוקנים בתוך ההקשר. המחברים מנתחים את התכונות של ייצוגי טוקנים תלוי הקשר הנוצרים על ידי הטרנספורמרים עלי ידי השוואתם עם ייצוגי הטוקנים המוזנים לבלוק הראשון של הטרנספורמר (כלומר ייצוגי הטוקנים ממטריצת embeddings של מודל שפה). השיפורים המוצעים במאמר באים למנוע מצב שבו ייצוגי תלוי ההקשר של טוקנים יהיו דומים מאוד אחד לשני.

תופעה דומה למתוארת בפסקה הקודמת נקראת over-smoothing ברשתות נוירונים גרפיות (GNN). זה קורה יש מספר גבוה מדי של שכבות MSA שמוביל לייצוגים דומים למדי של הקודקוד העלולים לגרום ל״קריסה״ של הייצוגים לתת-מחרב קטן של מרחב הקלט. ד״א מטריצת משקלי ה-attention בטרנספורמרים ניתן לראות בתור מטריצה שכנויות מנורמלת של גרף שלם.

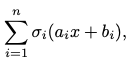
אבל איך נמדוד את מידת שוני (diversity) בין ייצוגי הטוקנים? המאמר מגדיר את שוני של מטריצה M(= קבוצה של וקטורים) בתור מינימום נורמת פרובניוס של ההפרש של M-A מעל כל המטריצות A בעלות רנק 1(כל הוקטורים במטריצה תלויים לינארית).

המחברים מראים כי עבור מודל המורכב מ l בלוקי MSA מוערמים (stacked) בלבד (ללא MLP) השוני של ייצוגי הפלט ניתן לחסום על מכפלה של הערכים הסינגולריים (הכללה של ערכים עצמיים למטריצות לא ריבועיות) המקסימליים של מטריצות משקלים השונות במנגנון MSA ובשוני של ייצוג הקלט (ממטריצת אמבדינגס של מודל השפה). ללא שכבות MLP ייצוגים אלו נוטים להתנוון ולהפוך להיות תלויים לינארית ככל מספר הבלוק l גדל. זו הסיבה ל״הימצאות״ של MLP בטרנספורמרים.

בנוסף עבור המודל המורכבים מבלוק MLP מוערמים המאמר מוכיח כי השוני של ייצוג הפלט הינם מכפלה של שוני ייצוג הקלט, הערכים הסינגולריים המקסימליים של מטריצות המשקלים וקבועי ליפשיץ של פונקציות האקטיבציה של MLP.

במטרה לשפר את תכונות ייצוגי הטוקנים בפלט של הטרנספורמר המאמר מציע שני שדרוגים, אחד ל MSA והשני ל-MLP. זוכרים בבלוק הטרנספורמר יש לנו יש לנו חיבור שארי (residual or shortcut according to the paper) - כלומר הפלט של MSA מחובר לייצוגי הקלט ל-MSA, המחברים מציעים לפתוח חיבורי ״קיצור דרך״ נוספים. כל חיבור כזה הוא למעשה שכבה לינארית עם מטריצה נלמדת ופונקצית אקטיבציה לא לינארית. כדי לא להכביד מדי על העומס החישובי המתווסף בעקבות כך(מטריצות המשקלים בחיבורי קיצור דרך אלו יכולות להיות 4096x4096 וזה די הרבה עם רוצה להשתמש בכמה חיבורים כאלו) משתמשים במטריצות בעלי רנק נמוך. המחברים מוכיחים שהוספה של שכבות לבלוקי הטרנספורמרים המקוריים אלו תורם להקטנת הפגיעה בשוני של ייצוגי פלט.

בנוסף המאמר מציע לשדרג פונקציה אקטיבציה שזה החלק המהותי של מנגנון הטרנספורמרים בנוסף ל-MSA. במקום להשתמש בפונקציה אקטיבציה רגיל (כמו סיגמאויד או ReLU) המאמר מציע לשלב (חיבורית) n פונקציות אקטיבציה בצורה הבא:

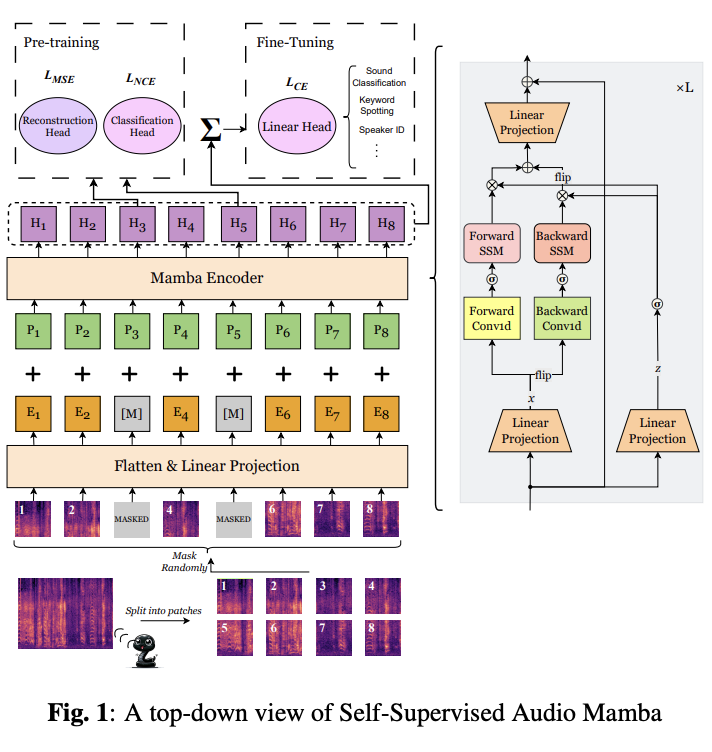


כאשר a\_i ו- b\_i הם פרמטרים נלמדים. כמובן שיש הוכחה שהחלפה כזו תורמת להגדלת השוני בין ייצוגי הפלט.

בנוסף השיפצורים המוצעים נבדקו על מספר בנצ'מארקים והראה ביצועים לא רעים.

## **Review 206: SSAMBA: SELF-SUPERVISED AUDIO REPRESENTATION LEARNING WITH MAMBA STATE SPACE MODEL**

https://arxiv.org/abs/2405.11831



המאמר הזה משך את תשומת ליבנו כי שמו דומה לממבה, ארכיטקטורה מעניינת שפרצה לתודעתנו לפני כחצי שנה וכבר יצאו עשרות מאמרים המשלבים אותה עבור מגוון דומיינים ומגוון משימות. והפעם התחום הוא אודיו והמחברים משתמשים בארכיטקטורת ממבה למטרת בניית ייצוג חזק של אות אודיו.

השאלה הראשונה שצריך לשאול כאן - מה הוא ייצוג חזק של דאטה. בהקשר זה באופן די טבעי ייצוג חזק של דאטה מקודד את התכונות החשובות שיש בדאטה כלומר דוחס את המידע המהותי שיש בדאטה בצורה יעילה. ייצוג זה נבנה על ידי מודל (מבוסס ממבה כאמור) ויכול לשמש אותנו לאימון של משימות נוספת על אותות אודיו. כלומר במקום לאמן מודל למשימה מסוימת על דאטה עצמו נאמן אותו על הייצוג הלטנטי של הדאטה (אמבדינג). דרך אגב התחום בלמידה מכונה העוסק בבנייה של ייצוגים אלו נקרא למידת הייצוג או representation learning.

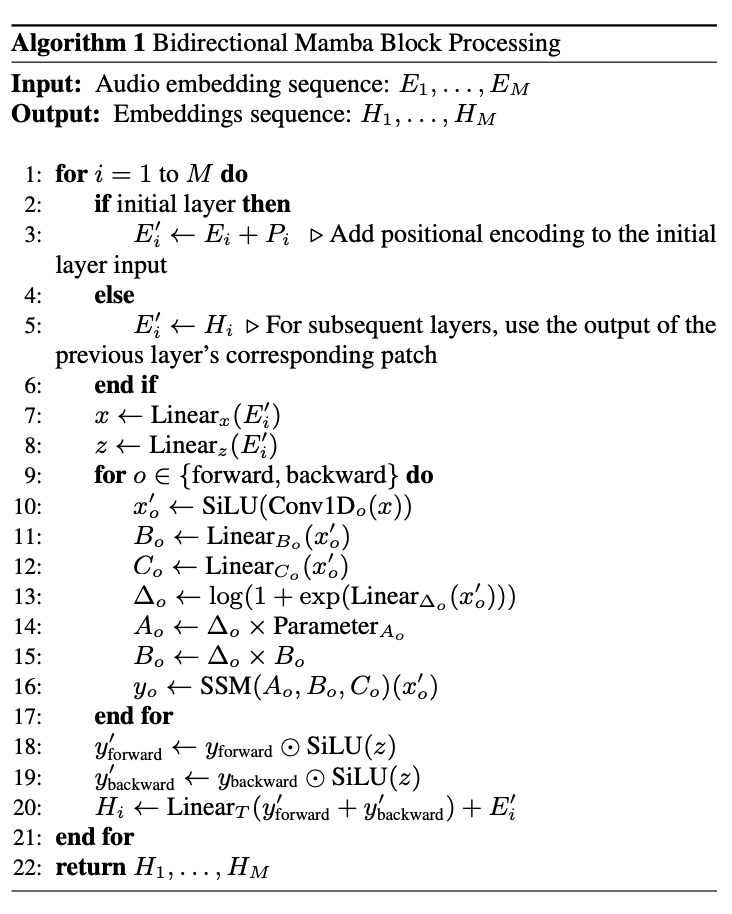
כמו שאתם בטח זוכרים ממבה אמור לקבל כקלט אמבדינגים של טוקנים. בשפה טבעית כל טוקן הוא תת-מילה או מילה מוגדרים על ידי המילון, עבור תמונה הטוקנים הם פאצ'ים של תמונה (בסדר מסוים) אבל מה אנו עושים עם אות האודיו? האמת משהו די סטנדרטי - מחלקים את האות שלנו למקטעים זרים שכל קטע הוא כמה שניות. לאחר מכן מעבירים כל מקטע כזה דרך התמרת פורייה ולאחר מכן דרך טרנספורמציית מל (Mel transform). בגדול טרנספורמציית מל מדגישה את התדרים שהאוזן האנושית מסוגלת לשמוע. לאחר מכן מעביר את התוצאה של מל דרך שכבה לינארית ומוסיפים קידוד מיקומי (positional encoding) המקודד מיקומו של כל טוקן אודיו בסדרה.

לאחר מכן מעבירים את התוצאה דרך שכבת ממבה (די סטנדרטית - ניתן למצוא את תיאורה בהרבה מקומות כולל בסקירותיי (**לינק)** הרבות בנושא זה). בדומה למודל ממבה לראייה ממוחשבת (שם המצב אפילו יותר מורכב כי הפאצ'ים של תמונה הם דו-מימדיים) כאן מכניסים את ייצוגי הטוקנים לממבה בשני ״סדרים״: מהתחלה עד הסוף (forward) ומהסוף להתחלה (backward) ומשלבים אותם כדי לבנות את הפלט.

מה שיוצא אחרי כמה שכבות של ממבה הוא למעשה ייצוג תלוי הקשר (contextualized) של הטוקן וכאמור ניתן לנצל אותו לאימון מודלים למגוון משימות ייעודיות.

אבל איך מאמנים את המודל המפיק את הייצוג הזה. בצורה די סטנדרטית האמת. ממסכים חלק מהטוקנים (כמו באימון של מודלי שפה) ואז בונים לוס המורכב משני חלקים:

1. הלוס הניגודי (contrastive loss): כאן המטרה לקרב את הייצוג של הטוקן הממוסך לייצוגו (מהאיטרציה הקודמת של אימון) ובאותו הזמן להרחיק אותו מהייצוגים של הטוקנים האחרים. ניתן להשיג את היעד הזה עם פונקציית לוס, לראשונה הוצגה במאמר InfoNCE (**לינק)** לפני 8 שנים בערך.
2. כאן מנסים לקרב את ייצוג הטוקנים הממוסכים עם ייצוגו (מהאיטרציה אימון הקודמת). המרחק בין חיזוי הייצוג והייצוג עצמו מוגדר כ L2 כלומר אוקלידי.



## **Review 206, Short: Training LLMs over Neurally Compressed Text**

[https://arxiv.org/pdf/2404.03626.pdf](https://arxiv.org/pdf/2404.03626.pdf?fbclid=IwAR3hisTocbuaG6aLOnkp219QEP8iJjQTVATjDJI7r-F5ZCvHcVR0EJevc4c_aem_AWuyyn8s2WSWRk9SvdjF64AAiDqZgA3Xz8oqDqBZlLv0cMKJT7iKLuWrxpJww1fe3tPyJkYSjGTO4Iv_uyUhTebi)

נתקלתי במאמר החמוד הזה של DeepMind and Anthropic

מה הוא בעצם מציע? לאמן מודל שפה לא על טקסט כמו שאנו רגילים היום אלא על טקסט מקומפרס. זה מגניב כי מודלי שפה ידועים ביכולתם לדחוס טקסט לייצוגים דחוסים אבל זה סיפור טיפה שונה.

אז מה בעצם נותן לנו אימון של llm על טקסטים דחוסים. קודם כל אימון מהר יותר, אורך הקשר ארוך יותר ויש עוד כמה. אז מה הבעיה? זה קצת עדין - הרי אם אנו דוחסים דאטה עם אלגוריתם חזק התוצאה תהיה רעש רנדומלי (אחרת המודל ילמד וינצל את זה).

אז מה המאמר בעצם עשה? הוא לקח מודל שפה M1 שאומן על סדרות ביטים שמייצגות את הטקסט ודחס את הפלט שלו. כמובן M1 גם דוחס את הדאטה (הרי זה מודל שפה) אבל לטענת המחברים בצורה רחוקה ממושלמת. אז הם לקחו שיטת דחיסה קלאסית הנקראת arithmetic coding(AC וכאמור דחסו את הפלט של M1. הם גם יצרו טוקנים חדשים אבל הפעם כל טוקן מיוצג על ידי צ'אנק של ביטים (באורך קבוע) הדוחס את של ביטי הקלט. כאן AC לוקח את ההסתבריות ש-M1 מוציא לכל טוקן ודוחס אותם. לאחר הפיכתם של סדרות אלו לטוקנים "הדחוסים״ מאמנים מודל שפה איתם בצורה הרגילה.

מעניין שניתוח ביצועים נעשה על ידי השוואות של perplexity המנורמל עם מקדם דחיסת דאטה (יודעים למה?). בסך הכל מאמר חמוד.