⚡️🚀המאמר היומי של מייק -07.11.24: ⚡️🚀

Cross-layer Attention Sharing for Large Language Models

אתם בטח יודעים הרצה של מודלי שפה עלול להיות דבר די יקר מבחינת משאבי חישוב וגם הזכרון. בטח כאשר יש לכם מודלים עם עשרות מיליארדי פרמטרים על עשרות רבות של שכבות של טרנספורמרים. אחד הדברים הכבדים שמצריכים לא מעט זיכרון הוא KV-Cache, שבו נשמרים המכפלות של ייצוגי (אמבדינגס) של הטוקנים במטריצות K ו- V לכל השכבות ולכל הטוקנים שכבר גונרטו (כולל הפרומפט - מדובר במודלי הדקודרים).

כמובן שכאשר המימדים של וקטורי הייצוג והמטריצות לא קטנים וגם אורך ההקשר נמדד בעשרות ומאות אלפים KV-Cache דורש הרבה מאוד זיכרון. בעבר יצאו לא מעט מאמרים שניסו לדחוס אותו על ידי ניתוח וזיהוי יתירויות אבל זה בד״כ נעשה פר שכבה (= בלוק הטרנספורמר). המאמר המסוקר מציע להתבונן בדחיסת KV-cache מפרספקטיבה רחבה יותר ולנסות לדחוס אותו דרך ניצול התלויות של ה-KV-cache בין השכבות השונות.

המחברים חקרו דמיון בין החלקים השונים בבלוק הטרנספורמרים (מכפלות של המטריצות השונות בוקטורי ייצוג, מקדמי attention וכדומה) והגיעו למסקנה שניתן ״להסיק״ את מקדמי ה-attention של שכבה n מהדאטה של שכבה n-1 בצורה חסכונית חישובית. כלומר עם הרבה פחות משקולות מהטרנספומר הרגיל. כלומר ההצעה היא לעשות סוג של LoRa אבל למקדמי ה-attention.

בצורה קצת יותר קונקרטית המאמר החליף מטריצות W\_Q ו-W\_K במטריצות בעלות ראנק נמוך (מכפלה של שתי מטריצות מלבניות כאשר המימד הפנימי של המכפלה נמוך - כלומר (M x k \* k x N) כאשר k קטן הרבה יותר מ- M ו- מ-M. מחשבים את הקלט לסופטמקס עם המטריצות האלו. לאחר מכן משרשרים אותם עם הקלט לסופטמקס מהשכבה הקודמת, מפעילים FFN והנה יש לנו קלט לסופטמקס בשכבה n. ושימו לב שאנו צריכים לשמור הרבה פחות דאטה ב- KV-cache כי יש לנו מטריצות בעלות ראנק נמוך.

איך מאמנים את הסיפור הזה? משלבים את הלוס הרגיל של מודל שפה עם לוס distillation שמטרתה לקרב את מקדמי -attention המחושבים בדרך המוצעת עם אלו שמחושבים עם מודל רגיל (עם attention ו- KV Cache רגילים).

מאמר די מעניין - אבל קצת ארוך מדי לדעתי אז תמצתתי לכם אותו 🙂

https://arxiv.org/abs/2408.01890