بسم الله الرحمن الرحیم

دانشکده مهندسی برق و کامپیوتر

پروژه درس هوش مصنوعی در سیستم‌های نهفته – بهمن ۱۴۰۲

پروژه

اهداف

هدف این تمرین، آشنایی با روال تبدیل مدل‌های شبکه‌های عصبی ترانسفورمر به باینری چند سطحی و ترکیب آن با هرس کردن پارامترهای مدل است.

۱) مقدمه

باینری سازی و هرس در شبکه‌های عصبی برای کاهش اندازه و پیچیدگی مدل‌های محاسباتی اهمیت زیادی دارند و باعث می‌شوند که این مدل‌ها برای استقرار، به ویژه در محیط‌های با منابع محدود، کارآمدتر باشند. باینری سازی شبکه را با کاهش دقت وزن‌ها ساده‌تر می‌کند، در حالی که حذف اتصالات کم اهمیت به پردازش سریع‌تر و کاهش استفاده از حافظه کمک می‌کند.

در ادامه چند نمونه از این روش ها را می‌بینید و یک جستجوی حالات روی این روش‌ها انجام می‌دهید تا تاثیر این روش ها را در دقت و اندازه مدل مشاهده کنید.

۲) محاسبات ماژول attention در ترانسفورمرها

ماژول self-attention

معماری‌های ترانسفورمر بر سازوکار خودتوجهی[۱] تکیه می‌کنند که در مقایسه با لایه‌های تکراری، قابلیت موازی سازی بهتری را برای مدل ایجاد می‌کند و نسبت به شبکه کانولوشنی نیاز کمتری به بایاس استدلالی دارد. این سازوکار به مدل امکان تمرکز روی بخش‌های مختلف دنباله ورودی را می‌دهد، همبستگی‌های دوبه‌دو را ایجاد کرده و وابستگی‌های برداری دوربرد بین عناصر دنباله ورودی را مدل کند. self-attention وزن‌های توجه را برای هر موقعیت در دنباله محاسبه می‌کند که نشان‌دهنده اهمیت هر موقعیت نسبت به دیگر موقعیت‌ها است که این

---

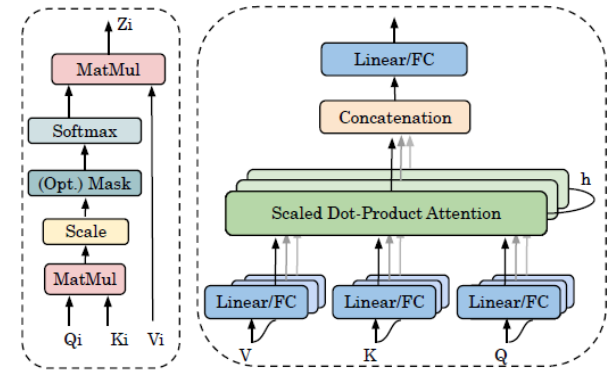
[۱] self-attention

ویژگی، این امکان را به مدل می‌دهد تا بسته به ورودی، به بخش‌های مختلف دنباله توجه کند. ورودی ماژول توجه به سه لایه کاملاً متصل (fully-connected) وارد می‌شود و برای تولید ماتریس‌های پرسش[2]، کلید[3] و مقدار[4] استفاده می‌شوند. وزن‌های این لایه‌های fully-connected در طول فرآیند آموزش یاد گرفته شده است. خروجی معادله (۱) نمایانگر تأثیر هر کلمه بر سایر کلمات است.

$$A = softmax\left(\frac{QK^T}{\sqrt{D_k}}\right) \qquad (۱)$$

عملیات ضرب نقطه‌ای بین پرسش و کلید به صورت جزء به جزء انجام می‌شود تا یک ماتریس امتیاز تولید شود، که بر $\sqrt{D_k}$ تقسیم می‌شود، و به این صورت مشکل ناپدید شدن گرادیان را کاهش می‌دهد. تابع Softmax مقادیری که امتیاز بالا دارند را افزایش و مقادیر با امتیاز پایین را کاهش می‌دهد. در نهایت، امتیاز توجه با ضرب ماتریس توجه و مقدار، همانطور که در معادله (۲) داده شده است، به دست می‌آید. نمایش طرح خودتوجهی در ۲ (الف) آورده شده است.

$$Attention\ (Q, K, V) = AV \qquad (۲)$$



شکل ۲- مکانیزم توجه در شبکه‌های ترانسفورمر[۷]

[2] Query
[3] Key
[4] Value

ماژول Multi-head self-attention

ماژول MHA شامل چندین "سر" است که هر کدام به طور همزمان عملیات توجه را محاسبه می‌کنند. همانطور که در شکل ۲(ب) نشان داده شده است، ورودی به ماژول MHA در تمامی سرها تکرار می‌شود. ورودی (X) به سر (سر i) از طریق سه لایه تماماً متصل پردازش می‌شود تا یک مجموعه از بردارهای پرسش (Q_i)، کلید (K_i) و مقدار (V_i) در هر سر، طبق معادله (۳) به دست آید.

$$Q_i = XW^{Q_i}, K_i = XW^{K_i}, V_i = XW^{V_i} \quad (۳)$$

خروجی هر سر(Z_i) از طریق سازوکار خودتوجهی و با استفاده از بردارهای Q_i K_i و V_i محاسبه می‌شود. معادله این موضوع را نشان می‌دهد.

$$head_i = Self - attention(Q_i, K_i, V_i), i = 1,2,..,h \quad (۴)$$

خروجی‌های مستقل از تمام سرها، طبق معادله (۵)، به صورت عمقی ترکیب و با استفاده از یک لایه تماماً متصل به فرم خطی تبدیل می‌شوند، تا خروجی ماژول MHA تولید شود.

$$MHA(Q, K, V) = [head_1; \dots head_h] * W^O \quad (۵)$$

شبکه FFN

لایه FFN یا پرسپترون چند لایه[5] شامل دو لایه کاملاً متصل با تابع فعال‌سازی ReLU یا GELU است. FFN اطلاعات وابسته به موقعیت را نسبت به مجموعه‌های مختلفی از دنباله‌های ورودی یاد می‌گیرد. خروجی MHA به FFNهای نقطه‌به‌نقطه وارد می‌شود که با استفاده از یک عملیات نرمال‌سازی[6] به طور بیشتری پردازش می‌شود.

---

[5] MLP

[6] Norm

روش‌های فشرده‌سازی

در این بخش سه روش هرس کردن و دو روش کوانتیزاسیون معرفی می‌شوند. هر گروه، از بین روش‌های هرس یکی را انتخاب کرده و انجام می‌دهد، و با توجه به روش هرس انتخابی جستجویی روی تکنیک‌های کوانتیزاسیون انجام خواهد داد.

روش اول هرس

در این روش، ابتدا ماتریس‌های پرسش و کلید، به ۴ بیت کوانتایز می‌شوند، سپس خروجی softmax محاسبه شده (A در معادله (۱)) و در نهایت عملیات top-k روی آن اعمال می‌شود. عمل کوانتیزاسیون در این روش، ترانکیشن است و تنها نیاز است که هربار، ۴ بیت بالای عدد حاصل جدا شود. عملیات top-k نیز بر حسب ورودی، درصدی از با اهمیت ترین اعداد (بیشترین امتیاز) را انتخاب کرده و باقی اعداد را صفر می‌کند. در نهایت با عملیات باینری، مدل تبدیل به باینری چند سطحی می‌شود.

توجه کنید که در این روش علاوه بر درصدهای مختلف هرس، باید عرض بیت‌های مختلف را نیز جستجو کنید.

روش دوم هرس

این روش اهمیت هر واژه یا عبارت ورودی (توکن) در یک بلوک attention را با استفاده از احتمالات آن ارزیابی می‌کند. توکن‌ها بر اساس اهمیت‌شان به سه سطح تقسیم می‌شوند. در ابتدا، مهم‌ترین توکن‌ها برای پردازش بیشتر انتخاب شده و بقیه حذف می‌شوند (۸۵ درصد مهم باقی می‌مانند و بقیه صفر خواهند شد). از میان این توکن‌های انتخاب شده، دسته‌بندی دومی اتفاق می‌افتد که در آن مهم‌ترین‌ها با دقت بالا به n بیت باینری شده و بقیه با دقت پایین‌تر (باینری m بیتی) پردازش می‌شوند (از میان ۸۵ درصد باقی مانده، ۷۰ درصد با اهمیت به n بیت باینری تبدیل شده، و ۳۰ درصد باقی به m بیت باینری تبدیل می‌شوند). توکن‌هایی که در مرحله اول حذف شده‌اند، به عنوان توکن‌های ۰ بیتی در نظر گرفته می‌شوند که به معنای حذف کامل آن‌ها است. این روش به منظور تعادل بین کارایی محاسباتی و دقت مدل، منابع را روی بخش‌های مهم‌تر داده‌ها متمرکز می‌کند.

توجه کنید که $n > m$ است. در این روش شما باید n و m را جستجو کنید.

روش سوم هرس

در این روش، عملیات top-k با درصدهای مختلف روی خروجی پرسش و کلید، جستجو و اعمال می‌شود. در نهایت با عملیات باینری، مدل تبدیل به باینری چند سطحی می‌شود.

توجه کنید که در این روش علاوه بر درصدهای مختلف هرس باید عرض بیت‌های مختلف را نیز جستجو کنید.

روش کوانتیزاسیون ممیز ثابت

در این روش، کوانتایزر با استفاده از یک ضریب (scaling factor) (برای کل یک ماتریس یا بردار)، که قابل یادگیری به همراه پارامترهای مدل است، مدل را با حفظ دقت به تعداد بیت کمتر کوانتایز می‌کند.

روش کوانتیزاسیون باینری چند سطحی

باینری چند سطحی، برای هر یک از مقادیر ماتریس‌ها، چند سطح (بیت) را درنظر می‌گیرد، با این تفاوت که نمایش اعداد بیشتری را نسبت به کوانتیزاسیون ممیز ثابت در بر می‌گیرد. برای هر یک از سطوح، یک ضریب (scaling factor) عدد ممیز شناور (یا ثابت) در نظر گرفته شده که به همراه پارامترهای مدل قابل یادگیری است.

پیش نیازهای انجام تمرین

- آشنایی اولیه با پایتان و شبکه‌های عصبی
- آشنایی با pytorch و شبکه‌های عصبی BERT در huggingface

برای اجرای کدها می توانید از colab استفاده کنید.

مراحل انجام تمرین

- هرگروه باید یکی از روش‌های هرس توضیح داده شده را انجام دهد. برای پیدا کردن روش هرس خود، دو رقم آخر شماره دانشجویی اعضای گروه جمع شده و باقی‌مانده آن بر سه + ۱، معادل روش هرس انتخابی آن گروه خواهد بود.
- کدهای مربوط به باینری و کوانتایز کردن هر لایه‌ی شبکه در اختیار شما قرار می‌گیرند. در دو پوشه rebnet و lsqplus این کدها وجود دارند. از تابع prepare موجود در هر یک برای تبدیل مدل خود به مدل باینری یا کوانتایز استفاده کنید. برای lsqplus از ۲lsqplus_quantize_V استفاده کنید.
- نتایج دقت و تابع خطا را برای هر یک از تنظیمات خود (درصد top-k، تعداد بیت‌های کوانتایز و ...) در یک جدول گزارش کنید.

لازم است موارد زیر جهت تحویل تمرین و ارائه‌ی گزارش رعایت شوند:

- گزارش خود را در بخش‌های مجزا شامل چکیده، نحوه‌ی انجام کار، نتایج به دست آمده، تحلیل نتایج، نتیجه‌گیری و ضمائم ارائه کنید. فایل گزارش باید بر اساس فرمت قرار داده شده در سایت درس باشد.
- درصورت استفاده از تکنیک‌های اضافه برای فشرده‌سازی، در گزارش توضیح دهید.
- فایل گزارش به صورت doc باشد. کد خود را نیز آپلود کنید.
- تمرین را با فرمت YourName_StudentNo_Project.rar آپلود کنید.
- گروه ها می‌تواند دو نفره باشند.
- بارگذاری فایل‌های گزارش توسط یکی از اعضای گروه کافی است.
- نمره از ۱۰۰ محاسبه می شود و به ازای هر روز تاخیر در اپلود تمرین، به اندازه 2^x که x تعداد روز تاخیر است از نمره شما کسر می‌شود.
- در صورت مشاهده تشابه زیاد در کدها و گزارش، نمره ۱۰۰- برای هر دو گروه اعمال خواهد شد.
- تمرین تحویل حضوری یا آنلاین دارد که زمان آن بعدا اعلام خواهد شد.

مراجع

[1] https://fastmachinelearning.org/hls4ml