مقاله "Optimizing High-Throughput Inference on Graph Neural Networks at Shared Computing Facilities with the NVIDIA Triton Inference Server"یاد گرفتم:

گزارش و برداشت از مقاله بهینه‌سازی استنتاج (Inference) با توان عملیاتی بالا در شبکه‌های عصبی گراف (GNN)با استفاده از سرور Triton انویدیا

سلام،

این مقاله به یک مشکل خیلی مهم و به‌روز در دنیای محاسبات علمی، به‌خصوص در جاهایی مثل فیزیک انرژی بالا (HEP) پرداخته. قضیه از این قراره که الگوریتم‌های یادگیری ماشین(ML) ، مخصوصا مدل‌های پیچیده‌ای مثل شبکه‌های عصبی گراف (GNN)، دارن روز به روز محبوب‌تر میشن چون نتایج فوق‌العاده‌ای میدن. مثلا تو فیزیک ذرات، از این مدل‌ها برای کارهایی مثل دسته‌بندی جت‌های ذرات استفاده می‌کنن که قبلا خیلی سخت بود.

مشکل اصلی چیه؟

این مدل‌های GNN خیلی قوی هستن ولی به همون اندازه هم محاسبات سنگینی دارن و بدون استفاده از پردازنده‌های گرافیکی (GPU)، اجرای اون‌ها) مرحله استنتاج یا (Inference خیلی کند میشه. از طرفی، خیلی از مراکز تحقیقاتی و دانشگاهی، مثل مرکز محاسباتی Fermilab که تو مقاله بهش اشاره شده، منابع محاسباتی رو به صورت اشتراکی در اختیار تعداد زیادی کاربر قرار میدن .(Shared Computing Facilities) مشکل اینجاست که اکثر منابع این مراکز، پردازنده‌های مرکزی (CPU) هستن و تعدادGPU ها محدوده و گرون هم هستن. در نتیجه، وقتی تعداد زیادی کاربر بخوان همزمان از مدل‌های ML سنگین استفاده کنن، هم سرعت کار میاد پایین) زمان رسیدن به نتیجه یا time-to-insight زیاد میشه (و هم از منابع بهینه استفاده نمیشه.

راه حل پیشنهادی مقالهNVIDIA Triton Inference Server :

اینجاست که مقاله یه راه حل جالب رو معرفی و تست می‌کنه: استفاده از نرم‌افزار متن‌باز NVIDIA Triton Inference Server.ایده اصلی اینه که به جای اینکه هر کاربر سعی کنه مدل ML خودش رو رویCPU های کند اجرا کنه یا منتظر بمونه تا یه GPU اختصاصی گیرش بیاد، یه سیستم مرکزی باGPU های قوی راه‌اندازی بشه که توسط Triton مدیریت میشه.

Tritonچطور کار می‌کنه؟ (چیزی که من فهمیدم)

Tritonمثل یه "مدیر برنامه" برای مدل‌های ML عمل می‌کنه.

۱. **مدیریت متمرکز :**مدل‌های ML مختلف) مثل همین ParticleNet که تو مقاله تست شده (روی این سرور مرکزی)که بهGPU ها وصله (قرار می‌گیرن.

۲. **درخواست از راه دور :**کاربرها کد اصلی تحلیل داده‌هاشون رو روی همونCPU های معمولی اجرا می‌کنن، اما هر وقت نیاز به اجرای مدل ML دارن، یه درخواست به سرور Triton می‌فرستن.

۳. **اجرا روی GPU:** سرور Triton این درخواست‌ها رو می‌گیره، داده‌ها رو به GPU می‌فرسته، مدل رو اجرا می‌کنه و نتیجه رو برای کاربر برمی‌گردونه.

٤. **سرویس‌دهی به چند کاربر :**Triton می‌تونه همزمان درخواست‌های چندین کاربر برای چندین مدل مختلف رو مدیریت کنه و از قابلیت‌هایی مثل **دسته‌بندی پویا (Dynamic Batching)** استفاده می‌کنه تا درخواست‌های کوچیک رو با هم ترکیب کنه و GPU رو حسابی مشغول نگه داره تا کارایی بالا بره.

٥. **مقیاس‌پذیری خودکار :(Auto-scaling)** می‌تونه تعداد نمونه‌های (instance) سرور رو بر اساس بار کاری (مثلا طول صف درخواست‌ها) کم و زیاد کنه تا هم به درخواست‌ها سریع جواب بده و هم منابع الکی هدر نره. تو مقاله نشون دادن که تنظیم درست پارامترهای این مقیاس‌پذیری چقدر روی کارایی تاثیر داره.

نتایج کلیدی و چیزهای جالبی که یاد گرفتم:

- **سرعت فوق‌العاده :**برای مدل ParticleNet که یه GNN نسبتا پیچیده است، استفاده از Triton روی GPU حدود **50برابر** سریع‌تر از اجرای همون مدل روی CPU بود! این یعنی صرفه‌جویی خیلی زیاد تو زمان. البته برای مدل‌های دیگه مثل ResNet50 این افزایش سرعت کمتر بود (حدود ۶ برابر) و برای مدل‌های خیلی ساده مثل درخت تصمیم(BDT) ، حتی کندتر هم شد **!نتیجه مهم :**همیشه باید تست کرد که آیا استفاده از Triton برای یک مدل خاص به‌صرفه هست یا نه.

- **عملکرد در شلوغی :**وقتی تعداد کاربرها (workers) که همزمان درخواست می‌فرستادن زیاد میشد، Tritonبا اضافه کردن نمونه‌های بیشتر) تا جایی که GPU خالی داشت (تونست توان عملیاتی (throughput) بالایی رو حفظ کنه. این نشون میده سیستم برای محیط‌های اشتراکی شلوغ خوب جواب میده.

- **چالش اجرای چند مدل همزمان :**اگه چند مدل مختلف بخوان همزمان روی یک نمونه (instance) از Triton اجرا بشن، ممکنه برای منابع) مثل حافظه (GPU رقابت کنن و کارایی هر کدوم پایین بیاد) بهش میگن .(thrashing راه بهتر اینه که اگه منابع اجازه میده، هر مدل روی نمونه جداگانه‌ای اجرا بشه. این مدیریت مدل‌ها (model orchestration)یه نکته مهمه که باید بهش توجه کرد.

- **فناوری :MIG** مقاله اشاره کرده که از قابلیت Multi-Instance GPU (MIG) درGPU های A100 استفاده کردن. این قابلیت اجازه میده یه GPU فیزیکی بزرگ به چندتا GPU مجازی کوچیک‌تر تقسیم بشه و هر کدوم به صورت مستقل به یه نمونه Triton اختصاص داده بشن که به مدیریت بهتر منابع کمک می‌کنه.

جمع‌بندی و اهمیت:

به نظرم این مقاله خیلی کاربردی بود چون یه راه حل عملی برای یه مشکل واقعی تو مراکز محاسباتی بزرگ ارائه میده. با استفاده از Triton میشه ازGPU های گرون‌قیمت به صورت بهینه و اشتراکی استفاده کرد تا سرعت تحلیل داده‌ها با مدل‌های ML

سنگین بالا بره و محقق‌ها بتونن سریع‌تر به نتایج علمی برسن. این الگو می‌تونه برای خیلی از مراکز محاسباتی دیگه که با چالش مشابهی روبرو هستن، مفید باشه. در واقع، این یه مثال خوب از پیاده‌سازی "استنتاج به عنوان سرویس (Inference-as-a-Service)" هست.