
بسم الله الرحمن الرحيم

دانشگاه اصفهان

دانشکده مهندسی کامپیوتر

گروه مهندسی نرم‌افزار

پروژه درس روش پژوهش و ارائه کارشناسی رشته‌ی مهندسی کامپیوتر گرایش هوش مصنوعی

استفاده از کانتینر ها برای اجرای سریع و مقیاس مدل های زبانی بزرگ

استاد راهنما:

دکتر زهرا زجاجی

دانشجو:

سهیل سلیمی

دی ۱۴۰۲

تقدیم به

خودم

چکیده

مدل های زبانی بزرگ [1] مدل هایی هستند که با استفاده از تکنیک های یادگیری عمیق بر روی داده های متنی بزرگ آموزش دیده اند و قادر به تولید متن های شبیه به انسان و انجام وظایف مختلف بر اساس ورودی ارائه شده هستند. این مدل ها می توانند برای تولید محتوای خلاق، ترجمه زبان ها، پاسخ به سوالات و انجام وظایف دیگر مورد استفاده قرار گیرند. اما اجرای این مدل ها در محیط های واقعی با چالش هایی مانند نیاز به منابع محاسباتی زیاد، حفظ حریم خصوصی و امنیت داده ها، و مسئولیت اخلاقی مواجه است. در این مقاله، ما یک روش برای استفاده از کانتینر ها برای اجرای سریع و محلی ممدل های زبانی بزرگ را ارائه می دهیم. کانتینر ها امکان ایجاد و اجرای محیط های نرم افزاری مستقل و قابل حمل را فراهم می کنند. ما نشان می دهیم که چگونه می توان با استفاده از کانتینر ها، مدل های زبانی را بدون نیاز به یک سرویس ابری، بر روی دستگاه های محلی اجرا کرد. ما مزایا و چالش های این روش را بررسی می کنیم و چندین مورد کاربردی را نشان می دهیم. ما نتایج آزمایش های خود را بر روی چندین مدل زبانی معروف و چندین وظیفه زبانی ارائه می دهیم و نشان می دهیم که این روش می تواند کارایی و دقت بالایی را حفظ کند. ما همچنین چندین جهت برای کارهای آینده در این زمینه پیشنهاد می دهیم.

کلیدواژه‌ها: ۱- کانتینر ها و مدل های زبانی بزرگ ۲- اجرای محلی و سریع مدل های زبانی ۳- بهینه سازی و امنیت مدل های زبانی ۴- مورد کاربردی مدل های زبانی بزرگ

[1] Large language model

فهرست مطالب

عنوان **صفحه**

عنوان **صفحه**

عنوان **صفحه**

فهرست تصاویر

عنوان **صفحه**

فهرست جداول

عنوان **صفحه**

فصل اول
مقدمه

۱-۱ پیش‌گفتار

مدل های زبانی بزرگ [1] [۱] در سال های اخیر توانایی شگفت انگیزی در وظایف پردازش زبان طبیعی و فراتر از آن نشان داده اند. این موفقیت مدل های زبانی بزرگ ها منجر به ورود تعداد زیادی از پژوهش های علمی در این زمینه شده است. این پژوهش ها موضوعات متنوعی را شامل می شوند، از جمله نوآوری های معماری، راهبردهای بهتر آموزش، بهبود طول متن، تنظیم دقیق، مدل های زبانی بزرگ های چند حالته، رباتیک، مجموعه داده ها، معیارهای ارزیابی، کارایی و بیشتر. با توجه به توسعه سریع روش ها و پیشرفت های مداوم در پژوهش مدل های زبانی بزرگ ها، درک تصویر کلی از پیشرفت ها در این راستا بسیار چالش برانگیز شده است.

در این مقاله، ما به بررسی چالش ها و راه حل های مربوط به اجرای سریع و محلی مدل های زبانی بزرگ ها می پردازیم. ما نشان می دهیم که چگونه می توان با استفاده از کانتینر ها، یک روش مدیریت بسته بندی و انتشار نرم افزار، مدل های زبانی بزرگ ها را بر روی رایانه های شخصی یا سرورهای خود اجرا کرد. این به معنای این است که نیازی به تکیه بر یک سرویس ابری برای استفاده از آنها نیست،

[1] Large language model

که می تواند چندین مزیت داشته باشد، از جمله: حفظ حریم خصوصی و امنیت داده ها: وقتی یک مدل های زبانی بزرگ محلی را اجرا می کنید، داده های شما هرگز از دستگاه شما خارج نمی شود. این می تواند برای داده های حساس، مانند سوابق بهداشتی یا داده های مالی، مهم باشد. در دسترس بودن آفلاین: مدل های زبانی بزرگ های محلی می توانند آفلاین استفاده شوند، که به این معنی است که شما می توانید از آنها حتی اگر اتصال اینترنت نداشته باشید، استفاده کنید. این می تواند برای کار بر روی پروژه های در مناطق دور افتاده یا برای برنامه هایی که نیاز به در دسترس بودن همیشگی دارند، مفید باشد. سفارشی سازی: مدل های زبانی بزرگ های محلی می توانند برای وظایف یا حوزه های خاص تنظیم دقیق شوند. این می تواند آنها را دقیق تر و کارآمدتر برای وظایفی که شما نیاز دارید انجام دهند، کند. مدل های زبانی بزرگ ها می توانند بر روی انواع پلتفرم های سخت افزاری، از جمله واحد پردازش مرکزی [۱] ها و واحد پردازش گرافیکی [۲] ها اجرا شوند. با این حال، مهم است توجه داشته باشید که مدل های زبانی بزرگ های محلی می توانند بسیار هزینه بر برای اجرا باشند، بنابراین شما ممکن است به یک رایانه قدرتمند برای استفاده از آنها به طور موثر نیاز داشته باشید. برای اجرای یک مدل های زبانی بزرگ محلی، شما باید نرم افزار لازم را نصب کنید و فایل های مدل را دانلود کنید. پس از انجام این کار، شما می توانید مدل را شروع کنید و از آن برای تولید متن، ترجمه زبان ها، پاسخ به سوالات و انجام وظایف دیگر استفاده کنید.

[1] CPU
[2] GPU

فصل دوم
مطالب اصلی

۲-۱ ادبیات موضوع
۲-۱-۱ کانتینر ها

کانتینر سازی یک روش مدیریت بسته بندی و انتشار نرم افزار است که به شکل مجازی برنامه های کاربردی را برای استقرار، بسته بندی و ایزوله می کند[۲]. کانتینرها از هسته سیستم عامل استفاده می کنند تا برنامه های کاربردی را از سخت افزار و سیستم عامل میزبان جدا کنند. این باعث می شود که برنامه ها قابل انتقال، سبک و سریع باشند.

کانتینر ها مفهومی هستند که به برنامه نویسان امکان می دهند تا برنامه های خود را به صورت مستقل و قابل انتقال بین محیط های مختلف اجرا کنند. کانتینر ها(شکل ۲-۱) را می توان با کانتینر کشی تشبیه کرد. کانتینر کشی یک روش حمل و نقل است که در آن کالاهای مختلف در جعبه های استاندارد قرار می گیرند و با استفاده از وسایل نقلیه مختلف مثل کشتی، قطار، کامیون یا هواپیما حمل می شوند. کانتینر های برنامه نویسی هم مثل جعبه های حمل و نقل، برنامه های مختلف را در خود جای می دهند و با استفاده از سیستم عامل های مختلف مثل لینوکس، ویندوز، مک یا اندروید اجرا می شوند. کانتینر ها از برنامه ها جدا هستند و فقط به منابع لازم برای اجرای آن دسترسی دارند. این

باعث می شود که برنامه ها سبک تر، سریع تر و امن تر باشند. کانتینر ها همچنین به برنامه نویسان امکان می دهند تا برنامه های خود را به راحتی به روز رسانی، تست، اشکال زدایی و توزیع کنند.

شکل ۲-۱ - نظم دهی کانتینر های به وسایل برای حمل

مثال ۲-۱ (کاربرد داکر). فرض کنید شما یک برنامه وب نوشته شده با پایتون دارید که از چندین کتابخانه و پکیج استفاده می کند. برای اجرای این برنامه، شما نیاز دارید که سیستم عامل، پایتون و تمام وابستگی های آن را نصب کنید. اگر شما بخواهید برنامه خود را به یک سرور دیگر منتقل کنید، شما باید همین کار را در آن سرور نیز تکرار کنید. این فرآیند زمان بر، خطا خیز و ناکارآمد است.

با استفاده از کانتینر ها، شما می توانید برنامه خود را به همراه تمام وابستگی های آن در یک فایل قابل حمل قرار دهید. این فایل را می توان به عنوان یک تصویر [۱] کانتینر نامید. سپس شما می توانید با استفاده از یک نرم افزار مدیریت کانتینر، مثل داکر [۲]، این تصویر را در هر سرور یا محصول دلخواه خود اجرا کنید. داکر مسئول این است که تصویر را به یک فرآیند در حال اجرا [۳] تبدیل کند و با سطح

[1] image
[2] Docker
[3] container

مناسب از جداسازی و امنیت، آن را در سیستم عامل میزبان قرار دهد. به این ترتیب، شما نگران نصب و پیکربندی وابستگی های برنامه خود در هر محصول نخواهید بود.

مدیریت کانتینر یک فرآیند است که به ایجاد، اجرا، مانیتورینگ، توقف و حذف کانتینرها می پردازد۳. برای مدیریت کانتینرها، نیاز به ابزارهایی است که به عنوان ارکستراسیون کانتینر شناخته می شوند. این ابزارها به مدیریت کانتینرهای متعدد بر روی یک یا چند سرور کمک می کنند. برخی از این ابزارها عبارتند از:

- Docker[3]: این ابزار یک پلتفرم کانتینر سازی است که به ساخت، اجرا و اشتراک گذاری کانتینرهای برنامه ای کمک می کند.
- Kubernetes[4]: این ابزار یک سیستم ارکستراسیون کانتینر اپن سورس و رایگان است که اولین نسخه های آن در کمپانی گوگل طراحی شد. این ابزار به مدیریت، مقیاس بندی و به روز رسانی کانتینرهای برنامه ای بر روی یک خوشه از سرورها کمک می کند.
- OpenShift: این ابزار یک پلتفرم کانتینر سازی تجاری است که بر پایه داکر و کوبرنتیس ساخته شده است. این ابزار به توسعه، اجرا و مدیریت کانتینرهای برنامه ای در محیط های ابری یا محلی کمک می کند.

کانتینرهای برنامه ای و کانتینرهای سیستمی دو نوع کانتینر هستند که بر اساس نوع برنامه های کاربردی که اجرا می کنند، تفاوت دارند. کانتینرهای برنامه ای، مثل داکر، فایل ها، وابستگی ها و کتابخانه های یک برنامه را برای اجرا در یک سیستم عامل کپسوله می کنند۲. این کانتینرها فقط یک برنامه را اجرا می کنند و نیازی به یک سیستم عامل مهمان ندارند. کانتینرهای سیستمی، مثل LXC یا LXD، یک سیستم عامل کامل را برای اجرا چندین برنامه در یک کانتینر کپسوله می کنند. این کانتینرها مانند یک ماشین مجازی عمل می کنند، اما با استفاده از هسته سیستم عامل میزبان به جای یک هایپروایزر.

۲-۱-۲ کانتینرها چه مزایایی دارند؟

برخی از مزایای کانتینرها عبارتند از:

- سرعت و کارایی: کانتینرها به دلیل حجم کم و استفاده بهینه از منابع سخت افزاری، سریع تر و کارآمدتر از ماشین های مجازی هستند. کانتینرها می توانند در چند ثانیه ایجاد، اجرا و حذف

شوند، در حالی که ماشین های مجازی ممکن است چند دقیقه زمان ببرند.

- انتقال پذیری و توزیع پذیری: کانتینرها می توانند بر روی هر دستگاهی که دارای نرم افزار کانتینر سازی است، اجرا شوند. این به این معنی است که شما می توانید یک کانتینر را بر روی یک رایانه شخصی، یک سرور، یک ابر یا یک دستگاه IoT اجرا کنید. همچنین، شما می توانید کانتینرها را به راحتی بین محیط های مختلف منتقل یا توزیع کنید.
- ایزولاسیون و امنیت: کانتینرها از یکدیگر و از سیستم عامل میزبان جدا هستند. این به این معنی است که اگر یک کانتینر دچار خرابی یا حمله شود، تاث

۲-۱-۳ مدل های زبانی بزرگ

مدل های زبانی بزرگ مدل هایی هستند که با استفاده از داده های متنی بسیار زیاد، قادر به تولید و درک متون در زمینه های مختلف هستند. این مدل ها از تکنیک های پیشرفته یادگیری عمیق استفاده می کنند و معمولا از چندین لایه شبکه عصبی تشکیل شده اند. برخی از مثال های مشهور از مدل های زبانی بزرگ عبارتند از: GPT-3، BERT، XLNet و T5. این مدل ها قابلیت های بسیار گسترده ای دارند، از جمله ترجمه، خلاصه سازی، تولید متن خلاقانه، پاسخ به سوالات و غیره. با این حال، این مدل ها نیز چالش ها و محدودیت هایی دارند، مانند نیاز به منابع محاسباتی زیاد، عدم قابل اعتماد بودن در برخی موارد و نگرانی های اخلاقی و حفظ حریم خصوصی.

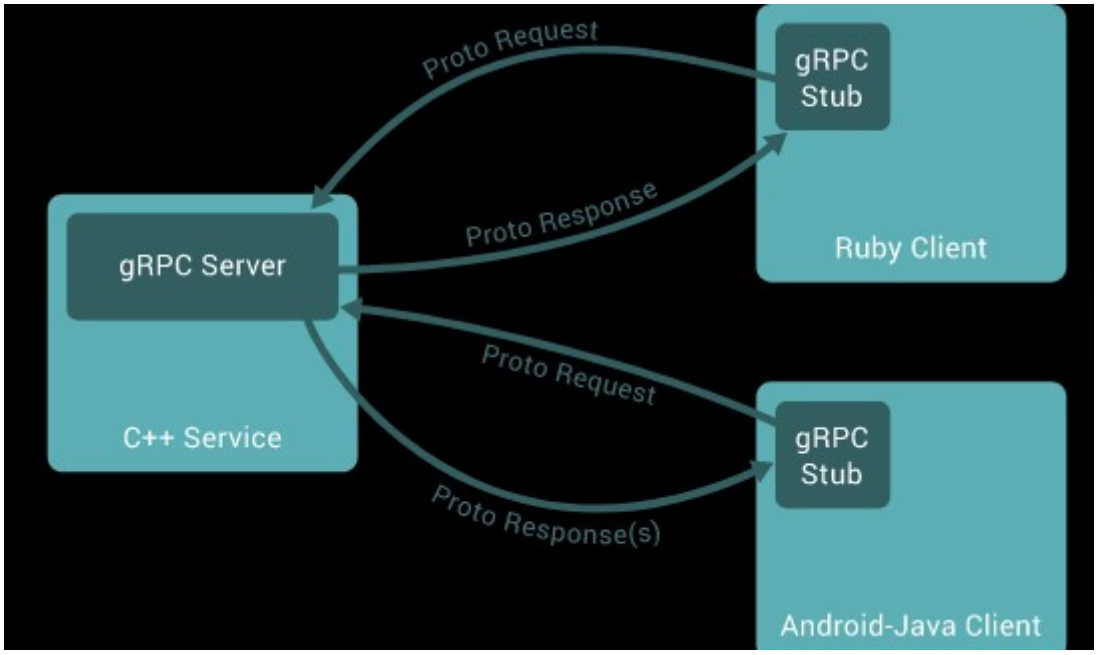
۲-۱-۴ پروتوکل gRPC

gRPC یک فریمورک مدرن و با کارایی بالا برای ارتباط بین سرویس‌ها است که از مفهوم Remote Procedure Call (RPC) استفاده می‌کند. در gRPC، سرویس‌ها می‌توانند با یکدیگر [۵]تعامل داشته باشند و توابع را از راه دور فراخوانی کنند. gRPC از زبان‌های مختلف برنامه‌نویسی پشتیبانی می‌کند و از پروتکل HTTP/2 برای انتقال داده‌ها استفاده می‌کند. gRPC از مزایای زیر برخوردار است:

- سرعت و کارایی: gRPC از فرمت سریال سازی Protocol Buffers استفاده می‌کند که یک فرمت دودویی، سبک و سریع است. این فرمت به gRPC اجازه می‌دهد تا داده‌ها را با حجم کمتر و سرعت بالاتر منتقل کند.
- تعریف قابل استفاده مجدد: gRPC از یک زبان تعریف سرویس (IDL) به نام proto3 استفاده

می‌کند که به شما اجازه می‌دهد تا تعریف سرویس خود را در یک فایل نوشته و آن را به زبان‌های مختلف تولید کنید. این به شما کمک می‌کند تا کد خود را قابل استفاده مجدد، خوانا و پایبند به قرارداد نگه دارید.

- پشتیبانی از جریان: gRPC از جریان دوطرفه پشتیبانی می‌کند که به شما اجازه می‌دهد تا داده‌ها را به صورت پشت سر هم و بدون درخواست-پاسخ منتقل کنید. این ویژگی به شما کمک می‌کند تا برای سناریوهای مختلف مانند چت، پخش زنده و رصد، از gRPC استفاده کنید.



شکل ۲-۲ - نحویه کار gRPC

طبق عکس ۲-۲، نحوه کار gRPC را با یک سرویس C++ و کلاینت‌هایی به زبان Ruby و Android-Java نشان می‌دهد. در اینجا، gRPC سرور، مجهز به یک سرویس C++، درخواست‌های Proto را از کلاینت‌ها دریافت می‌کند و با پاسخ‌های Proto پاسخ می‌دهد.

روند ارتباط بین gRPC سرور و کلاینت‌ها به شرح زیر است:

هر کلاینت یک gRPC Stub را ایجاد می‌کند که یک شیء است که متدهای سرویس را تعریف می‌کند و به آدرس gRPC سرور متصل می‌شود. هر کلاینت یک یا چند درخواست Proto را با استفاده از gRPC Stub به gRPC سرور می‌فرستد. درخواست Proto یک پیام است که با پروتوباف تعریف شده است و داده‌های مورد نیاز برای فراخوانی متد سرویس را حاوی است. gRPC سرور درخواست Proto را

دریافت می‌کند و آن را به متد مربوطه در سرویس C++ منتقل می‌کند. سرویس C++ منطق کسب و کار خود را اجرا می‌کند و یک پاسخ Proto را تولید می‌کند. پاسخ Proto یک پیام است که با پروتوباف تعریف شده است و نتیجه فراخوانی متد سرویس را حاوی است. gRPC سرور پاسخ Proto را به gRPC Stub کلاینت می‌فرستد. RPC Stub پاسخ Proto را به زبان کلاینت تبدیل می‌کند و آن را به کلاینت ارائه می‌دهد.

۲-۲ روش های پیشین

۲-۲-۱ استفاده از ماشین های مجازی

ماشین مجازی یک نرم افزار است که به شما اجازه می دهد یک سیستم عامل دیگر را در داخل سیستم عامل فعلی خود اجرا کنید. برای استفاده از مدل های زبانی بزرگ ها، شما نیاز دارید که یک ماشین مجازی با سیستم عامل ویندوز را نصب کنید و سپس نرم افزار مدل های زبانی بزرگ را در آن اجرا کنید.[۶] این روش دارای برخی مزایا و معایب است. برخی از مزایای استفاده از ماشین مجازی عبارتند از:

- شما می توانید از مدل های زبانی بزرگ ها بدون نیاز به خرید یک کامپیوتر ویندوز استفاده کنید.
- شما می توانید به راحتی بین سیستم عامل های مختلف جابجا شوید و فایل های خود را به اشتراک بگذارید.
- شما می توانید تنظیمات و پیکربندی های مختلف را برای ماشین مجازی خود انجام دهید و در صورت لزوم به حالت قبل بازگردانید.

برخی از معایب استفاده از ماشین مجازی عبارتند از:

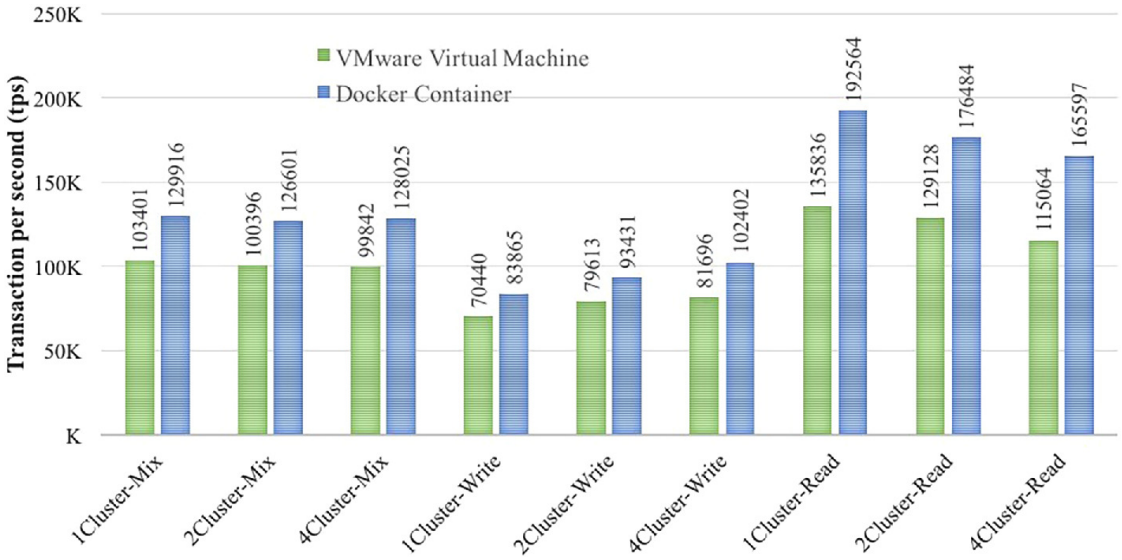
- شما نیاز دارید که فضای حافظه و پردازنده کافی را برای اجرای ماشین مجازی فراهم کنید، در غیر این صورت سرعت و عملکرد آن کند خواهد شد.
- شما نیاز دارید که یک نسخه قانونی از سیستم عامل ویندوز را تهیه و فعال کنید، در غیر این صورت با مشکلات قانونی و امنیتی روبرو خواهید شد.
- شما نمی توانید از برخی قابلیت های سخت افزاری کامپیوتر خود، مانند دوربین، صدا، چاپگر و غیره، در محیط مجازی استفاده کنید، مگر اینکه درایور های مناسب را نصب کنید.

۲-۲-۲ مقایسه کانتینرها و ماشین های مجازی

مقایسه این دو در جدول ۲-۱ امده است. که واضح است برای پروژه ما کانتینر ها بهینه تر و مناسب تر هستند.

جدول ۲-۱ - مقایسه کانتینرها و ماشین های مجازی

ویژگی	ماشین مجازی (VM)	کانتینر
جداسازی	از سیستم عامل میزبان و ماشین‌های مجازی دیگر کاملاً جدا می‌شود. این مورد زمانی مفید است که مرز امنیتی قوی ایجاد شود	از سیستم عامل میزبان و کانتینرهای دیگر به صورت سبک جدا می‌شود، اما مرز امنیتی به اندازه ماشین مجازی قوی نیست
سیستم عامل	یک سیستم عامل کامل از جمله هسته را اجرا می‌کند و بنابراین منابع سیستم بیشتری (CPU، حافظه و ذخیره‌سازی) را مصرف می‌کند	بخش حالت کاربر سیستم عامل را اجرا می‌کند و می‌تواند به گونه‌ای سفارشی شود که فقط خدمات مورد نیاز برنامه را شامل شود
سازگاری مهمان	می‌تواند هر سیستم عاملی را درون ماشین مجازی اجرا کند	باید با نسخه سیستم عامل میزبان هماهنگ باشد
مجازی‌سازی	سیستم کامپیوتری را مجازی‌سازی می‌کند، یعنی لایه‌های سخت‌افزاری	سیستم عامل را مجازی‌سازی می‌کند، یعنی فقط لایه‌های نرم‌افزاری
اندازه	اندازه ماشین مجازی بسیار بزرگ است، معمولاً در مقیاس گیگابایت	اندازه کانتینر بسیار سبک است، معمولاً چند صد مگابایت، اگرچه ممکن است بسته به کاربرد متفاوت باشد
زمان اجرا	ماشین مجازی زمان بیشتری برای اجرا می‌برد تا کانتینر، زمان دقیق بستگی به سخت‌افزار زیرین دارد	کانتینر زمان خیلی کمتری برای اجرا می‌برد
حافظه	ماشین مجازی حافظه زیادی را مصرف می‌کند	کانتینر حافظه بسیار کمی را می‌طلبد
امنیت	ماشین مجازی امن‌تر است، زیرا سخت‌افزار زیرین بین فرآیندها به اشتراک گذاشته نمی‌شود	کانتینر کمتر امن است، زیرا مجازی‌سازی بر پایه نرم‌افزار است و حافظه بین فرآیندها به اشتراک گذاشته می‌شود
کاربرد	ماشین‌های مجازی زمانی مفید هستند که ما نیاز داریم تمام منابع سیستم عامل را برای اجرای برنامه‌های مختلف استفاده کنیم	کانتینرها زمانی مفید هستند که ما نیاز داریم حداکثر برنامه‌های در حال اجرا را با استفاده از سرورهای حداقلی اجرا کنیم



شکل ۲-۳ - مقایسه استفاده منابع [۷]

۲-۲-۳ سرویس های ابری

سرویس های ابری را می توان برای استفاده از مدل های زبانی بزرگ ها به عنوان یک راه حل مقیاس پذیر و انعطاف پذیر در نظر گرفت. با استفاده از سرویس های ابری، می توان از منابع محاسباتی و ذخیره سازی بدون نگرانی از محدودیت های سخت افزاری بهره برد. همچنین، می توان با استفاده از سرویس های ابری، مدل های زبانی بزرگ ها را به صورت خودکار و پویا مدیریت کرد و به روز رسانی کرد[۸].

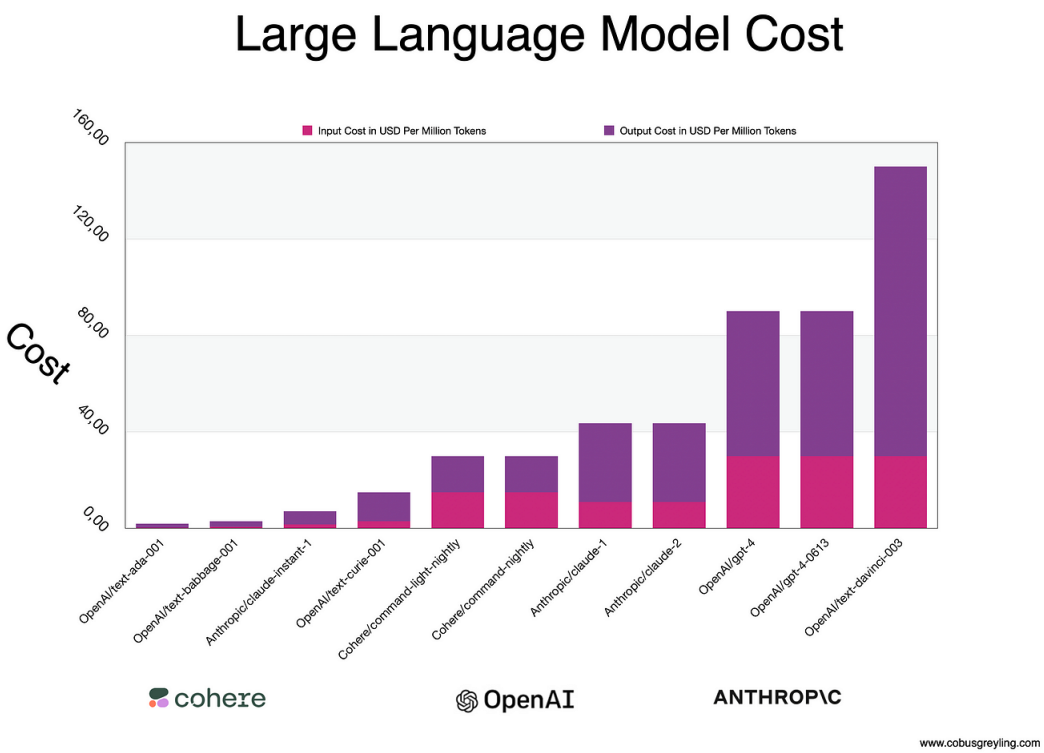
با این حال، استفاده از سرویس های ابری نیز مشکلات خود را دارد. برخی از مشکلات عبارتند از:

- حفظ امنیت و حریم خصوصی داده ها و مدل های زبانی بزرگ ها در فضای ابری
- تضمین کیفیت سرویس و عملکرد مناسب مدل های زبانی بزرگ ها در شرایط نامطلوب شبکه
- هزینه بالای استفاده از سرویس های ابری برای برخی از کاربردهای مدل های زبانی بزرگ ها
- عدم وجود استانداردهای یکسان و قابل تبادل بین سرویس دهندگان مختلف ابری

جدول ۲-۲ - هزینه استفاده از سرویس های ابری Open Ai [۹]

Output	Input	Model
$0.06/ 1K tokens	$0.03/ 1K tokens	gpt-4
$0.12/ 1K tokens	$0.06/ 1K tokens	gpt-4-32k

با توجه به جدول ۲-۲ برای تولید ۱۰۰۰ توکن شما باید حدود $ 58 پرداخت کنید.



شکل ۲-۴ – مقایسه هزینه ها برای اسفاده از سرویس های ابری[۱۰]

۲-۲-۴ استفاده مستقیم از مدل های زبانی بزرگ

برای استفاده مستقیم از مدل های زبانی بزرگ نیاز مشکلات زیر را به همراه دارد

- نیاز به فرد متخصص
- عدم مقیاس پذیری
- امکان استفاده ان در سیستم عامل ها مختلف وجود ندارد

۲-۳ روش پیشنهادی

۲-۳-۱ استفاده از کانتینر ها

کانتنر ها راهی برای بسته بندی و اجرای برنامه های کامپیوتری هستند که می توانند در محیط های مختلف اجرا شوند. کانتنر ها مزایایی مانند سادگی، قابلیت حمل و نقل، امنیت و کارایی دارند. برای انتشار مدل های زبانی بزرگ، کانتنر ها می توانند راه حل مناسبی باشند. چون:

- کانتنر ها می توانند مدل ها را به صورت جداگانه و مستقل از سخت افزار و سیستم عامل اجرا کنند. این به این معنی است که مدل ها را نیازی نیست برای هر پلتفرم یا دستگاه جدید تغییر داد یا تطبیق داد.
- کانتنر ها می توانند مدل ها را به صورت خودکار و پویا مقیاس بزرگ کنند. این به این معنی است که بر اساس نیاز و تقاضای کاربران، تعداد و منابع کانتنر ها را می توان افزایش یا کاهش داد.
- کانتنر ها می توانند مدل ها را به صورت امن و قابل اعتماد اجرا کنند. این به این معنی است که کانتنر ها محافظت شده از دسترسی های غیرمجاز یا خطای سخت افزار یا نرم افزار هستند.

برای استفاده از کانتنر ها برای انتشار مدل های زبانی بزرگ، لازم است چند قدم را طی کنیم:

- ابتدا باید یک تصویر [۱] کانتنر را بسازید. تصویر کانتنر شامل کدهای، پکیج های، پیکربندی های و داده های لازم برای اجرای مدل است.
- سپس باید تصویر کانتنر را در یک رجیستر [۲] آپلود کنید. رجیستر یک سرویس ذخیره سازی است که تصویر کانتنر را در دسترس قرار می دهد.
- در نهایت باید یک نمونه [۳] از تصویر کانتنر را در یک سرویس حمل و نقل [۴] درخواست کنید. سرویس حمل و نقل چگونگی و کجای اجرای نمونه را تعیین می کند.

ولی این مراحل دسترسی سریع را برای کاربر اینجا نمی کند. چون همچنان کار با این نوع سیستم ساخته شده مشکل است.

۲-۳-۲ مدیریت کانتنر های ساخته شده برای مدل های زبانی بزرگ

- برای اجرای مدل مدل های زبانی بزرگ ، ما از فناوری کانتینر استفاده می‌کنیم که با استانداردهای Open Container Initiative [۱۱] سازگار است. این فناوری به ما امکان می‌دهد که مدل را به صورت جدا و مستقل از سیستم عامل و محیط اجرایی بسته‌بندی و اجرا کنیم.
- برای اجرای کانتینر مدل مدل های زبانی بزرگ ، ما از یک اجراکننده کانتینر به نام youki [۱۲]

[1] image
[2] registry
[3] instance
[4] transport

استفاده می‌کنیم که یک پیاده‌سازی کامل از استاندارد OCI است. این اجراکننده کانتینر به ما امکان می‌دهد که کانتینر را با سرعت و امنیت بالا اجرا کنیم.

- برای تنظیم وابستگی‌های مورد نیاز مدل مدل های زبانی بزرگ ، ما یک فایل تنظیمات به فرمت JSON ایجاد می‌کنیم که شامل اطلاعاتی مانند نام کانتینر، نسخه مدل، پارامترهای مدل، حافظه مورد نیاز، پورت‌های مورد استفاده و دیگر تنظیمات مربوطه است. این فایل تنظیمات به اجراکننده کانتینر ارسال می‌شود تا کانتینر را با توجه به آن ایجاد و اجرا کند.
- برای ارتباط با مدل مدل های زبانی بزرگ ، ما از دو پروتوکل و gRPC پشتیبانی می‌کنیم. پروتوکل HTTP یک پروتوکل مبتنی بر درخواست-پاسخ است که از طریق وب ارتباط برقرار می‌کند. پروتوکل gRPC یک پروتوکل مبتنی بر RPC است که از طریق پروتوکل HTTP/2 ارتباط برقرار می‌کند. این دو پروتوکل به ما امکان می‌دهند که با استفاده از فرمت‌های مختلفی مانند JSON، XML، Protobuf [۱۳] و غیره، داده‌ها را به مدل ارسال و از مدل دریافت کنیم. برای تنظیم پروتوکل‌ها و تنظیمات مربوطه، ما از فایل‌های تعریف سرویس و تنظیمات استفاده می‌کنیم که شامل اطلاعاتی مانند نام سرویس، نوع درخواست، نوع پاسخ، پورت‌ها، مسیرها و دیگر جزئیات مربوطه است. این فایل‌ها به اجراکننده کانتینر ارسال می‌شوند تا پروتوکل‌ها و تنظیمات را برای کانتینر فعال کند.
- و برای کاربران عادی نیز که نیاز به API´ [۱۴] ندارند یک صفحه وب تهیه می شود که می توانند با ان به راحتی با مدلی که نصب کرده اند ارتباط برقرار کنند.

۲-۳-۳ چرا این روش برای کاربران عادی بهتر است؟

استفاده از کانتینرها برای اجرای سریع و محلی مدل‌های زبانی بزرگ می‌تواند برای یک کاربر عادی مفید باشد، زیرا:

- کانتینرها از منابع سیستم کمتری نسبت به ماشین‌های مجازی استفاده می‌کنند و بنابراین می‌توانند سرعت و کارایی بالاتری داشته باشند.
- کانتینرها امکان اجرای مدل‌های زبانی بر روی CPU را فراهم می‌کنند، که ممکن است برای کاربرانی که GPU ندارند یا محدودیت‌های هزینه‌ای دارند مفید باشد.
- کانتینرها امکان اجرای مدل‌های زبانی بدون نیاز به اتصال به اینترنت را می‌دهند، که می‌تواند

برای حفظ حریم خصوصی و امنیت کاربران مهم باشد.

- کانتینرها امکان انتخاب و تغییر مدل‌های زبانی را به راحتی فراهم می‌کنند، که می‌تواند برای انجام وظایف مختلف و تنظیم پارامترهای مدل مفید باشد.

۲-۳-۴ چرا این روش برای شرکت ها بهتر است؟

مزایای استفاده از کانتینرها برای شرکت ها:

- کاهش هزینه‌ها: با استفاده از کانتینرها، می‌توان مدل‌های زبانی بزرگ را بر روی سخت‌افزارهای محلی اجرا کرد، بدون نیاز به استفاده از سرویس‌های ابری یا اینترنت. این کار می‌تواند هزینه‌های مربوط به پردازش، ذخیره‌سازی و انتقال داده‌ها را کاهش دهد.
- افزایش امنیت: با استفاده از کانتینرها، می‌توان مدل‌های زبانی بزرگ را در محیط‌های ایزوله و کنترل شده اجرا کرد، بدون نیاز به اشتراک‌گذاری داده‌ها یا مدل‌ها با سرویس‌های خارجی. این کار می‌تواند خطرات مربوط به نشت اطلاعات، سوءاستفاده از مدل‌ها یا حملات سایبری را کاهش دهد.
- افزایش سرعت: با استفاده از کانتینرها، می‌توان مدل‌های زبانی بزرگ را با زمان راه‌اندازی کمتر و بازدهی بالاتر اجرا کرد، به دلیل بسته‌بندی قبلی برنامه‌ها و وابستگی‌ها. این کار می‌تواند تجربه کاربری را سریع‌تر و پاسخگوتر کند.
- افزایش مقیاس‌پذیری: با استفاده از کانتینرها، می‌توان مدل‌های زبانی بزرگ را به راحتی بر اساس تقاضا بزرگنمایی یا کوچکنمایی کرد.

فصل سوم
نتیجه‌گیری و پیشنهادها

۳-۱ نتیجه‌گیری

‘ در این مقاله، ما نشان دادیم که استفاده از کانتینر ها برای اجرای سریع و محلی مدل های زبانی بزرگ مزایای قابل توجهی دارد. کانتینر ها به ما امکان می دهند تا مدل های زبانی را بدون نیاز به نصب پیش نیاز های پیچیده و تنظیمات سخت افزاری، در هر سیستم عامل و پلتفرمی اجرا کنیم. این کار باعث افزایش قابلیت استفاده، کارایی و امنیت مدل های زبانی می شود. همچنین، کانتینر ها به ما کمک می کنند تا مدل های زبانی را به راحتی به صورت توزیع شده و مقیاس پذیر در شبکه های ابری یا لوکال اجرا کنیم. در نهایت، کانتینر ها به ما اجازه می دهند تا مدل های زبانی را با استفاده از فن آوری های جدید و بهینه سازی شده برای عملکرد بالاتر، بروز رسانی و توسعه دهیم. بنابراین، استفاده از کانتینر ها برای اجرای سریع و محلی مدل های زبانی بزرگ یک روش جذاب و قابل اعتماد است که در آینده بسیار پرکاربرد خواهد بود. برخی دیگر از مزایای این روش به صورت زیر است:

- با استفاده از کانتینرها، ما می‌توانیم مدل مدل های زبانی بزرگ را در هر محیطی که دارای اجراکننده کانتینر باشد، به راحتی اجرا کنیم. این کانتینرها سرعت و کارایی بالایی دارند و نیاز

به نصب و تنظیمات اضافی ندارند.

- این روش هیچ هزینه ای برای کاربران ندارد. کاربران فقط کافی است کانتینر مدل مدل های زبانی بزرگ را دانلود و اجرا کنند و از آن بهره ببرند. همچنین کاربران می‌توانند کانتینر را با دیگران به اشتراک بگذارند و یا از کانتینرهای دیگران استفاده کنند.
- این روش امنیت داده‌های کاربر را حفظ می‌کند. کاربران نیازی ندارند که داده‌های خود را به سرورهای خارجی یا ابری ارسال کنند و یا از سرویس‌های پرداختی استفاده کنند. کانتینر مدل مدل های زبانی بزرگ روی رایانه یا شبکه داخلی کاربر اجرا می‌شود و داده‌ها در دسترس کاربر می‌مانند.
- این روش به ما امکان می‌دهد که به یک API واحد برای مدل های زبانی دسترسی پیدا کنیم. ما می‌توانیم از پروتوکل‌های HTTP و gRPC برای ارتباط با مدل مدل های زبانی بزرگ استفاده کنیم و داده‌ها را با فرمت‌های مختلفی مانند JSON، XML، Protobuf و غیره ارسال و دریافت کنیم. این API واحد ما را از پیچیدگی‌های مربوط به مدل‌های زبانی مختلف مستقل می‌کند.

۳-۲ پیشنهادها

یکی از چالش های موجود در استفاده از مدل های زبانی بزرگ، نیاز به منابع محاسباتی زیاد و پیچیده است. این مدل ها معمولاً نیاز به تعداد زیادی از پردازنده های گرافیکی یا تنسوری دارند که همه آنها باید با هم همگام سازی شوند. این کار باعث می شود که اجرای این مدل ها در محیط های محلی یا کوچک بسیار دشوار و گران قیمت باشد. برای حل این مشکل، یک راه حل ممکن استفاده از کانتینر هاست. کانتینر ها روشی برای بسته بندی و اجرای نرم افزار ها به صورت جدا و مستقل از سخت افزار و سیستم عامل زیرین هستند. با استفاده از کانتینر ها، می توان یک محیط یکنواخت و قابل حمل برای اجرای مدل های زبانی بزرگ فراهم کرد. در این مقاله، ما پروژه ای ساختم که هدف عمده ان روی اجرای سریع محلی برای کاربر عادی یا سرور ها بود ولی مقایس پذیری این کانتنر ها در این مقاله بررسی نشده. و می توان این مورد را هم بررسی دقیق تر نمود که چگونه کانتنر های خود را بتوانیم روی چندین سرور در یک شبکه محلی قرار دهیم [۱۵].

منابع و مآخذ

[1] B. Min, H. Ross, E. Sulem, A. P. B. Veyseh, T. H. Nguyen, O. Sainz, E. Agirre, I. Heintz, and D. Roth, "Recent advances in natural language processing via large pre-trained language models: A survey," *ACM Computing Surveys*, vol.56, no.2, pp.1–40, 2023.

[2] Y. Hayut, "Containerization and the load center concept," *Economic geography*, vol.57, no.2, pp.160–176, 1981.

[3] I. Docker, "Docker," *lınea].[Junio de 2017]. Disponible en: https://www. docker. com/what-docker*, 2020.

[4] T. Kubernetes, "Kubernetes," *Kubernetes. Retrieved May*, vol.24, p.2019, 2019.

[5] X. Wang, H. Zhao, and J. Zhu, "Grpc: a communication cooperation mechanism in distributed systems," *SIGOPS Oper. Syst. Rev.*, vol.27, p.75–86, jul 1993.

[6] O. Tickoo, R. Iyer, R. Illikkal, and D. Newell, "Modeling virtual machine performance: challenges and approaches," *ACM SIGMETRICS Performance Evaluation Review*, vol.37, no.3, pp.55–60, 2010.

[7] S. Shirinbab, L. Lundberg, and E. Casalicchio, "Performance evaluation of containers and virtual machines when running Cassandra workload concurrently," *Concurrency and Computation: Practice and Experience*, vol.32, 2 2020.

[8] Z. Li, H. Zhang, L. O'Brien, R. Cai, and S. Flint, "On evaluating commercial cloud services: A systematic review," *Journal of Systems and Software*, vol.86, no.9, pp.2371–2393, 2013.

[9] "Open ai pricing," Jan 2024.

[10] C. Greyling, "How does large language models use long contexts?,"

[11] R. Girma, "Evaluation of container virtualization systems supporting open container initiative images," 2018.

[12] "containers/youki," Jan 2024.

[13] S. Popić, D. Pezer, B. Mrazovac, and N. Teslić, "Performance evaluation of using protocol buffers in the internet of things communication," in *2016 International Conference on Smart Systems and Technologies (SST)*, pp.261–265, IEEE, 2016.

[14] J. Bloch, "How to design a good api and why it matters," in *Companion to the 21st ACM SIGPLAN symposium on Object-oriented programming systems, languages, and applications*, pp.506–507, 2006.

[15] M. A. Tamiru, J. Tordsson, E. Elmroth, and G. Pierre, "An experimental evaluation of the kubernetes cluster autoscaler in the cloud," in *2020 IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*, pp.17–24, IEEE, 2020.