HelixLLM

HelixLLM

از لپ‌تاپ تا کلاستر Anthropic و OpenAI یک باینری، شش حالت — استنتاج سازگار با چند میزبانه.

خلاصه

HelixLLM یک سیستم توزیع‌شده LLM در سطح سازمانی است که بر پایه Go ساخته شده است: یک باینری واحد با سیستمی از حالت‌ها که از توسعه تک‌میزبانه تا تولید چندمیزبانه مقیاس‌پذیر را از طریق Anthropic و OpenAI است. این سیستم‌رابط‌های برنامه‌نویسی کاملاً سازگار با زنجیره جایگزین چندارائه‌دهنده با llama.cpp، ارائه می‌دهد، با استنتاج محلی HTTP/3 و سیستم عامل RAG امتیازدهی، پایپ‌لاین ReAct.

توضیح کوتاه

HelixLLM یک سیستم توزیع‌شده LLM تک‌باینری و مبتنی بر Go است. این سیستم‌رابط‌های HTTP/3را از طریق Anthropic و OpenAI در دسترس قرار می‌دهد، برنامه‌نویسی سازگار با llama.cpp استنتاج محلی را اجرا می‌کند، ارائه‌دهندگان ابری‌رایگان‌را به‌طور خودکار شناسایی و به همراه RAG امتیازدهی کرده و در زنجیره جایگزین قرار می‌دهد، و علاوه بر این، پایپ‌لاین دانش عامل ReAct با قابلیت فراخوانی ابزاررا اضافه می‌کند — قابل استقرار در شش حالت مختلف.

توضیح مفصل

HelixLLM یک سیستم توزیع‌شده LLM با Go و Gin ساخته شده و ترفند اصلی آن این است که یک مصنوع واحد برای تمام مقیاس‌ها خدمت‌رسانی می‌کند. این سیستم به یک باینری واحد کامپایل می‌شود که سیستم حالت‌های آن در زمان استقرار تصمیم می‌گیرد این باینری برای نمونه‌ای همه‌کاره روی لپ‌تاپ اجرا کنید، یا مسئولیت‌ها `full` چه نقشی ایفا کند: آن‌را به صورت تقسیم کرده و روی چندین `control` و `agents`، `knowledge`، `brain`، `gateway`را بین حالت‌های میزبان توزیع کنید — همان کد، با چینش متفاوت به جای بازنویسی، از دستگاه توسعه‌دهنده تا کلاستر تولیدی.

این سیستم به دو گویش مسلط است:رابط‌های برنامه‌نویسی کاملاً سازگار با OpenAI و Anthropic، موجود از هر دو اکوسیستم بدون تغییر کار می‌کنند SDK به طوری که کلاینت‌های نسخه ۱.۳ ارائه TLS و HTTP/2 با بازگشت خودکار به HTTP/3 (QUIC) همه از طریق ROCm و Metal، CUDA با پشتیبانی از llama.cpp می‌شوند. استنتاج محلی از طریق

انجام می‌شود، بنابراین همان بیلد روی سخت‌افزارهای انویدیا، اپل و ای‌ام‌دی به‌طور یکسان شتاب می‌گیرد. ویژگی برجسته آن زنجیره جایگزین چندارائه‌دهنده است که ناپایداری معروف استنتاج ابری به‌طور خودکار مدل‌های HelixLLM زایگان‌را به منبعی مدیریت‌شده و خودترمیم‌شونده تبدیل می‌کند (Chutes، OpenRouter، HuggingFace، Nvidia، Cerebras، SambaNova، Together) رایگان‌را از بیش از ۷ ارائه‌دهنده ابری شناسایی می‌کند، هر ۵ دقیقه آن‌هارا از طریق زنجیره رتبه‌بندی‌شده با جایگزینی LLMsVerifier امتیازدهی کرده و درخواست‌هارا از طریق زنجیره رتبه‌بندی‌شده با جایگزینی LLMsVerifier طریق محلی به عنوان آخرین راهکار llama.cpp هدایت می‌کند — همیشه با xxخودکار خطاهای ۴۲۹/۵ تضمین‌شده، به طوری که هیچ درخواستی صرفاً به دلیل عدم دسترسی به ارائه‌دهنده شکست نخورد.

RAG (جذب، HelixLLM، فراتر از استنتاج خام) یک پلتفرم کامل کاربردی است: پایپ‌لاین دانش با قابلیت فراخوانی ابزار، جلسات ReAct و سیستم عامل (vector قطعه‌بندی، تعبیه، جستجوی در همان باینری گنجانده شده‌اند. سیستم حالت‌ها در سطح شبکه نیز RAG مکالمه و یکپارچگی با سربار Go تمام لایه‌ها از طریق فراخوانی‌های مستقیم درون‌فرایندی full سودمند است — در حالت شبکه صفر ارتباط برقرار می‌کنند، در حالی که همان باینری، زمانی که روی میزبان‌های مختلف توزیع Brotli/ هماهنگ می‌شود. در پایان، مذاکره محتوا با Kafka و gRPC، SSE می‌شود، از طریق بایت‌به‌بایت مطابقت دارد، احراز Anthropic و OpenAI که با فرمت‌های SSE استریم، gzip، ردیابی Prometheus، همراه با محدودیت نرخ، معیارهای JWT و API هویت با کلید تکمیل‌کننده این Go و مجموعه بزرگی از زیرماژول‌های زیرساختی تولید OpenTelemetry سیستم هستند.

محتوا

چرا آن‌را ساختیم

تیم‌ها به استنتاجی نیاز دارند که قابل حمل، سازگار با استانداردها و مقاوم باشد—بدون نیاز به بازنویسی به گونه‌ای طراحی شد که HelixLLM. کلاینت‌ها یا وابستگی به یک ارائه‌دهنده یا یک ماشین خاص همان باینری بتواند به صورت محلی برای توسعه اجرا شود و در عین حال به یک خوشهٔ تولیدی چند که کلاینت‌ها از پیش استفاده Anthropic و OpenAI میزبانه مقیاس‌پذیر باشد، و با گویش‌های می‌کنند، ارتباط برقرار کند.

چرا یک تغییردهندهٔ بازی است

و عامل‌ها—را در یک RAG، این ابزار کل پشتهٔ استنتاج—درگاه، استنتاج محلی، بازگشت به ابر باینری واحد فشرده می‌کند که با یک کلید حالت کنترل می‌شود، به طوری که معماری‌ای که مستقر می‌کنید یک تصمیم زمان اجرا است نه یک پروژه بازطراحی پلتفرم. و چیزی‌را که پیش‌تر یک نقطه ضعف بود به یک قابلیت تبدیل می‌کند: قابلیت اطمینان ارائه‌دهندگان ابری به یک دغدغهٔ درجه یک و پیوسته اندازه‌گیری‌شده بدل می‌شود که توسط زنجیره‌ای خودترمیم‌شونده و امتیازدهی‌شده مدیریت می‌شود؛ زنجیره‌ای که هر چند دقیقه یک بار ارائه‌دهندگان‌را دوباره رتبه‌بندی می‌کند و همواره به استنتاج محلی تضمین‌شده تنزل می‌یابد. قابلیتی که این امر آزاد می‌کند، یک نقطهٔ پایانی واحد است که واقعاً می‌توانید به آن تکیه کنید—سازگار با استانداردها، قابل حمل از لپ‌تاپ تا خوشه، و ناتوان از خاموشی به دلیل محدودیت نرخ یا شکست یک ارائه‌دهندهٔ بالادستی.

نوآوری‌ها

- یک باینری واحد با سیستمی شش‌حالته که به صورت یکپارچه یا در نقش‌های توزیع‌شده اجرا gRPC/SSE/Kafka و full، درون‌فرایندی در حالت Go می‌شود—فراخوانی‌های مستقیم

در حالت‌های توزیع‌شده—به طوری که توپولوژی استقرار بدون تغییر کد یا هزینه‌های شبکه‌ای ناخواسته تغییر می‌کند.

- زنجیره‌ای خودکشف‌شونده و امتیازدهی‌شده از ارائه‌دهندگان چندگانه با بیش از ۷ ارائه‌دهندهٔ رایگان رتبه‌بندی می‌شوند، با شکست خودکار در برابر LLMsVerifier که به طور پیوسته توسط به عنوان آخرین راهکار—تبدیل llama.cpp و بازگشت تضمین‌شده به xxخطاهای ۴۲۹/۵ ظرفیت لایهٔ رایگان به ظرفیتی قابل اعتماد.
- ارائه HTTP/3 که از طریق Anthropic و OpenAI سطوح سازگار با هر دو استاندارد به طوری که کلاینت‌های هر دو اکوسیستم بدون نیاز HTTP/2، می‌شوند، با بازگشت خودکار به به تغییر متصل می‌شوند.
- —در یک کدبیس واحد پشتیبانی می‌کند ROCm و CUDA، Metal استنتاج محلی که از همان بیلد روی سخت‌افزارهای انویدیا، اپل و ای‌ام‌دی با شتاب اجرا می‌شود.

بزرگ‌ترین چالش‌های فنی و راه‌حل‌های ما

- بیشتر سیستم‌ها مرز سخت‌گیرانه‌ای **مقیاس‌پذیری از یک میزبان به چند میزبان بدون بازنویسی.** بین «توسعه محلی» و «تولید توزیع‌شده» قائل می‌شوند و عبور از این مرز به معنای بازطراحی معماری است. ما این مرز را با سیستمی حالت‌محور در یک باینری واحد حذف کردیم: همان لایه‌ها از طریق فراخوانی‌های مستقیم درون‌فرایندی ارتباط برقرار می‌کنند و به طور شفاف `full` در حالت سوئیچ می‌کنند، به طوری که مقیاس‌پذیری gRPC/SSE/Kafka در حالت‌های توزیع‌شده به تنها یک تغییر پیکربندی است نه یک انتقال.
- استنتاج لایهٔ رایگان تا زمانی که با **ارائه‌دهندگان ابری رایگان غیرقابل اعتماد و محدود به نرخ** خطای ۴۲۹ مواجه نشود یا در میانهٔ درخواست ناپدید نشود، سریع است. ما آن‌را قابل اعتماد ردیابی پیشگیرانهٔ LLMsVerifier، ساختیم با کشف خودکار مدل‌های موجود، امتیازدهی آن‌ها با هزهای محدودیت نرخ برای دور زدن ارائه‌دهندگانی که در آستانهٔ محدودسازی هستند، و شکست محلی—به طوری که ناپایداری این llama.cpp خودکار به پایین زنجیرهٔ رتبه‌بندی‌شده تا مجموعه هرگز به کلاینت نمی‌رسد.
- بازنویسی کلاینت‌ها برای پذیرش یک بک‌اند استنتاجی جدید **سازگاری کلاینت در دو اکوسیستم.** تا جزئیات—Anthropic و OpenAI مربوط به API غیرممکن است. ما هر دو شکل های هر دو اردوگاه SDK متمایز آن‌ها—را پیاده‌سازی کردیم، به طوری که SSE فرمت‌های استریم اشاره می‌کنند و به سادگی کار می‌کنند HelixLLM به.

محتوا

پشته فناوری

- **Go + Gin** — انتخاب شده زیرا یک باینری واحد با اولویت همزمانی، چیزی است که کل سیستم حالت‌ها را ممکن می‌سازد: یک بیلد که می‌تواند هم به‌عنوان سرور لپ‌تاپ و هم نقش دروازه‌را در خود جای می‌دهد HTTP کلاستر عمل کند. این باینری کل سیستم و لایه.
- **HTTP/3 (QUIC) + TLS ۱٫۳، با بازگشت به HTTP/2** — انتخاب شده برای انتقال مدرن، کم‌تأخیر و مقاوم در برابر قطع اتصال، که به‌عنوان سطح سرور با مذاکره خودکار ارائه HTTP/2 استفاده کنند، به‌صورت خودکار به QUIC می‌شود تا کلاینت‌هایی که نمی‌توانند از بازگردند.
- **llama.cpp (CUDA/Metal/ROCm)** — انتخاب شده برای استنتاج محلی قابل‌حمل که روی بک‌اندهای انویدیا، اپل و ای‌ام‌دی از یک کدبیس واحد شتاب می‌گیرد؛ همچنین به‌عنوان تأمین‌کننده تضمینی آخرین راهکار عمل می‌کند تا زنجیره بازگشت هرگز به بن‌بست نرسد.
- **LLMsVerifier** — انتخاب شده تا پرسش «کدام تأمین‌کننده در حال حاضر خوب است»را به عدد تبدیل کند؛ این ابزار زنجیره بازگشت ابری‌را هر پنج دقیقه امتیازدهی و رتبه‌بندی می‌کند تا مسیریابی بر اساس کیفیت زنده انجام شود، نه فرضیات قدیمی.

- **تأمین‌کنندگان ابری (Chutes، OpenRouter، HuggingFace، Nvidia، Cerebras، SambaNova، Together)** — انتخاب شده برای بهره‌برداری از ظرفیت لایه‌رایگان در چندین منبع بالادستی؛ به‌صورت خودکار کشف و در یک زنجیره بازگشت واحد رتبه‌بندی می‌شوند تا هیچ تأمین‌کننده‌ای نقطه شکست نباشد.
- **gRPC + SSE + Kafka** — انتخاب شده به‌عنوان پروتکل‌های انتقال بین‌حالته برای استقرارهای توزیع‌شده: gRPC برای تماس‌های سرویس‌به‌سرویس، SSE برای جریان‌سازی و Kafka برای جریان رویدادهای غیرهمزمان بین نقش‌ها.
- **embeddings / پایگاه داده برداری** — انتخاب شده برای تقویت خط لوله دانش RAG از ابتدا تا انتها: دریافت، قطعه‌بندی، تعبیه و جستجو در اسناد برای پایه‌گذاری پاسخ‌های مدل.
- **Prometheus + OpenTelemetry** — انتخاب شده برای معیارها و ردیابی توزیع‌شده که یک درخواست‌را در هر حالتی که مستقر شده باشد دنبال می‌کند.
- **vasic-digital Go زیرماژول‌های** — انتخاب شده برای استفاده مجدد از زیرساخت‌های تولیدی سخت‌شده به‌جای بازسازی آن‌ها، تا پایه سیستم با پشته گسترده‌تر هماهنگ بماند.

وضعیت و یادداشت‌های شفافیت

- **وضعیت: بتا.** سیستم استنتاج توزیع‌شده کاربردی و در حال توسعه فعال.
- **مجوز: نامشخص.** مخزن در متادیتای خود هیچ مجوزی اعلام نکرده است (تهی `licenseInfo`) — این وضعیت تأیید نشده است و باید پیش از اعلام مجوز حل شود.
- مخزن مرجع در حال حاضر به `github.com/HelixDevelopment/llm` ارجاع می‌دهد؛ به آن تغییر مسیر می‌دهد. آمار آستانه پوشش و تعداد زیرماژول‌ها در فایل `HelixLLM` مسیر خوداظهاری شده است README.

اولویت سطح: Helix-اصلی.