ریاست جمهوری
معاونت علمی، فناوری و اقتصاد دانش بنیان

مهندسی پرامپت در عصر هوش مصنوعی عامل‌محور (Agentic AI)

منبع: سایت TechTarget | نویسنده: بث پاریسو

خلاصه‌مطلب

مهندسی پرامپت، فراتر از یک واژه پرکاربرد، به یک مهارت ضروری برای توسعه‌دهندگان نرم‌افزار تبدیل شده است، به‌ویژه با ظهور "هوش مصنوعی عامل‌محور " (Agentic AI). این نسل نوین هوش مصنوعی شامل عوامل خودمختاری است که قادر به اجرای مستقل وظایف پیچیده، حتی نوشتن و استقرار کامل برنامه‌ها هستند.

این تحول، نقش اجرا کننده را، از صرفاٍ کدنویسی به ترکیبی از برنامه‌نویسی، منطق، تعامل با زبان طبیعی و تخصص در معماری سیستم‌های هوش مصنوعی تغییر می‌دهد و نیازمند تسلط همزمان بر مهارت‌های سنتی و جدید است.

بهترین شیوه‌ها برای بهره‌گیری از این فناوری در حال شکل‌گیری است و شامل تکنیک‌هایی مانند تفکیک وظایف، استفاده از خروجی‌های ساختاریافته، تولید افزوده بازیابی (RAG) و یادگیری چند نمونه‌ای می‌شود. با این حال، نظارت و قضاوت انسانی برای انتخاب ابزار مناسب، ارزیابی نتایج، بهینه‌سازی هزینه‌ها و مهم‌تر از همه، تضمین ایمنی و قابلیت اطمینان سیستم‌های عامل، همچنان حیاتی است.

ایجاد محیط‌های آزمایشی امن برای آزمایش و درک رفتار این سیستم‌های قدرتمند و گاه غیرقابل پیش‌بینی، یکی از راهکارهای کلیدی در این مسیر است.

برای مشاهده متن اسکن کنید

متن کامل

بهترین شیوه‌های عملی در شرکت‌هایی مانند Oracle، LinkedIn و AWS در حال شکل‌گیری است، و هوش مصنوعی عامل‌محور در آستانه‌ی دگرگون ساختن کار روزمره‌ی توسعه‌دهندگان نرم‌افزار قرار دارد.

مهندسی پرامپت (Prompt Engineering) از زمانی که هوش مصنوعی زایشی (Generative AI) پدیدار شد، به واژه‌ای پرکاربرد تبدیل شده است، اما با ورود به مرحله‌ی جدیدی از هوش مصنوعی، اهمیت آن دوچندان شده است.

مهندسی پرامپت به مجموعه‌ای از تکنیک‌ها گفته می‌شود که برای بهینه‌سازی نتایج مدل‌های زبانی بزرگ (Large Language Models یا LLMs) استفاده می‌شود؛ مدل‌هایی که در هسته‌ی ابزارهای هوش مصنوعی زایشی قرار دارند. LLMهایی مانند ChatGPT موج اول فناوری GenAI بودند که اواخر ۲۰۲۲ وارد بازار فناوری سازمانی شدند. طی ۱۸ ماه گذشته، نسل جدیدی از GenAI با عنوان هوش مصنوعی عامل‌محور (Agentic AI) پدیدار شده است؛ جایی که اجزای نرم‌افزاری خودگردان با پشتیبانی از LLMها به‌طور مستقل جریان‌های کاری چندمرحله‌ای را اجرا می‌کنند.

در حالی که مهندسی پرامپت می‌تواند پاسخ‌های GenAI را قابل‌استفاده‌تر و دقیق‌تر کند، این مهارت در جایی که LLMها شروع به اقدام می‌کنند (مانند نوشتن و استقرار کامل یک اپلیکیشن) حیاتی می‌شود.

هوش مصنوعی عامل‌محور، یک لایه‌ی جدید از انتزاع بین توسعه‌دهندگان انسانی و کد ایجاد می‌کند. اگرچه همچنان ممکن است برنامه‌نویسی در کار باشد، اما این فرایند شامل رشته‌هایی مانند منطق، هنر و زبان طبیعی

نیز می‌شود. این امر توسعه‌ی اپلیکیشن را برای طیف گسترده‌تری از افراد قابل دسترسی می‌سازد و در عین حال، زمینه‌ای برای تخصص جدیدی به‌نام مهندسی هوش مصنوعی (AI Engineering) فراهم می‌آورد؛ تخصصی که برای توسعه‌ی پیچیده‌ترین و ظریف‌ترین سیستم‌های مبتنی بر هوش مصنوعی ضروری است.

با این حال، به گفته‌ی متخصصان باتجربه، اغلب توسعه‌دهندگان نرم‌افزار باید میان مهارت برنامه‌نویسی و مهارت در مهندسی پرامپت تعادل برقرار کنند تا در دنیای جدید عامل‌های هوش مصنوعی، نقش خود را حفظ کنند.

کریشنان سریدهار (Krishnan Sridhar)، معاون مهندسی در LinkedIn مي گوید: «اگر بحث ساخت اپلیکیشن‌های ساده باشد، یک عامل (Agent) می‌تواند آن را بسازد. اما برای نرم‌افزارهای بزرگ سازمانی، همچنان به انسان نیاز داریم تا جنبه‌های کلیدی را طراحی کند.»

ساخت زیربنای اتوماسیون با هوش مصنوعی

هوش مصنوعی عامل‌محور، گامی به جلو در اتوماسیون فناوری اطلاعات است که در آن سیستم‌ها خود تصمیم می‌گیرند چه کاری انجام دهند. اما یک اصل قدیمی در علوم کامپیوتر همچنان برقرار است:

"ورودی نامناسب یعني خروجی نامناسب" (Garbage in, garbage out).

سریدهار می‌گوید: «ما متوجه شدیم اتوماسیون در جایی بهترین عملکرد را دارد که سیستم‌ها ساده و قابل پیش‌بینی طراحی شده باشند. در نهایت، تمام مدل‌های زبانی با زبان طبیعی آموزش دیده‌اند و تلاش می‌کنند کد

را مانند انگلیسی بنویسند. بنابراین، توصیف واضح معماری سیستم و نام‌گذاری منابع، در تصمیم‌گیری دقیق عامل‌ها نقش اساسی دارد.»

مهندسی پرامپت: از تجربه تا استاندارد

پس از ایجاد زیرساخت اولیه، توسعه‌دهندگان LinkedIn شروع به کار با LLMهای منفرد کردند. از این تجربیات اولیه، برخی بهترین شیوه‌ها (Best Practices) شکل گرفته‌اند.

کارتیک رامگوپال (Karthik Ramgopal)، مهندس ارشد در LinkedIn، می‌گوید: «یکی از یافته‌های ما این است که مدل‌ها محدودیت‌های شناختی دارند. برای رسیدن به کیفیت بهتر، باید وظایف را به بخش‌های کوچک‌تر تقسیم کرد و پرامپت‌ها را به‌صورت زنجیره‌ای طراحی کرد. این روش، امکان آزمودن قطعه‌به‌قطعه را فراهم می‌کند و کیفیت را بالا می‌برد.»

همچنین، مدل‌های جدیدتر LLM می‌توانند خروجی‌های ساختاریافته مانند JSON یا XML تولید کنند، که مطابق با انتظار اپلیکیشن‌های پایین‌دستی است. این قابلیت، روش‌های قدیمی مانند «حلقه‌های بازخورد و اصلاح» را غیرضروری می‌سازد.

رامگوپال تأکید می‌کند که توسعه‌دهندگان باید با تکنیک‌هایی مانند تولید تقویت‌شده با بازیابی (Retrieval-Augmented Generation یا RAG) و یادگیری چندمثاله (Few-Shot Learning) آشنا شوند. این تکنیک‌ها برای تغذیه‌ی مؤثرتر داده به مدل‌ها به‌کار می‌روند.

او می‌افزاید: «همه‌ی این تصمیمات همچنان بر عهده‌ی توسعه‌دهنده

است: آیا از زنجیره تفکر (Chain of Thought) استفاده کنم یا درخت تفکر (Tree of Thoughts)؟ داده‌های نمونه را چگونه فراهم کنم؟ آیا آن‌ها را از پایگاه داده بردارم یا دستی بنویسم؟»

ابزار درست برای کار مناسب

به گفته آنتیه بارت (Antje Barth)، کارشناس ارشد در AWS، این مسیر همچنان نیازمند تجربه‌ی انسانی برای ارزیابی و هماهنگی سیستم‌های هوش مصنوعی است. او می گوید: «هوش مصنوعی عامل‌محور همیشه قطعی نیست — ممکن است چند پیشنهاد مختلف ارائه دهد. اینجاست که تجربه‌ی توسعه‌دهنده در انتخاب بهترین مسیر اهمیت پیدا می‌کند.»

از سوی دیگر، سوده راغاوان (Sudha Raghavan) از Oracle هشدار می‌دهد که باید هزینه‌های رایانشی و منابع ابری برای اجرای GenAI را در نظر گرفت. می گوید: «لزومی ندارد برای یک مسئله ساده در چت‌بات، خوشه‌ای هزار گره‌ای از GPUها راه‌اندازی کنید. تفکر توسعه‌دهنده از کدنویسی به طراحی سریع و کم‌هزینه تغییر کرده است.»

همچنین، مهندسی پرامپت نقش مهمی در مدیریت هزینه و زمان پاسخ دارد. برای مثال، جای‌گذاری بخش‌های ثابت پرامپت در ابتدای متن باعث می‌شود Cache پیشوندی (Prefix Caching) فعال شده و بهره‌وری افزایش یابد.

سطح بالاتری از منطق

پس از آشنایی توسعه‌دهندگان با LLMها، گذار به هوش مصنوعی عامل‌محور

نیازمند تغییری دوباره در رویکرد است. به گفته بارت، توسعه‌دهندگان می‌توانند عامل‌ها را همانند طراحی پرامپت، با جزئیات دقیق و منطق مرحله‌ای هدایت کنند.

بارت مي گويد: «در Amazon Q Developer، اجزای مختلفی وجود دارد: کمک به تست واحد، فاز برنامه‌ریزی، تولید کد درون‌خط. همه‌ی این‌ها با هدف تبدیل وظایف عامل‌محور به ابزار قدرتمند واحد طراحی شده‌اند.»

در آینده، با پیشرفت LLMها و توانایی استدلال آن‌ها، توسعه‌دهندگان باید دانش خود را به معماری سیستم‌ها گسترش دهند.

یان بیور (Ian Beaver)، دانشمند ارشد داده در Verint می‌گوید: «مهندسی پرامپت همچنان مهارتی کلیدی خواهد بود، زیرا راهی است برای توصیف وظایف و منابع در سیستم‌های عامل‌محور.»

ظهور کارگران دیجیتال

به گفته اندی تورای (Andy Thurai) از Constellation Research، در آینده توسعه‌دهندگان می‌توانند وظایف پیچیده‌ای مانند تحلیل داده یا عملیات پشتیبانی را به عوامل دیجیتال بسپارند.

در این نقطه، تأکید بیشتر بر طراحی تجربه‌ای ایمن، شخصی‌سازی‌شده و قابل اعتماد برای کاربران نهایی خواهد بود.

بارت می گوید: «این مهارت‌های معماری هستند که توسعه‌دهندگان باید برای ساخت تجربه‌های مبتنی بر داده‌ی سازمانی در هوش مصنوعی عامل‌محور یاد بگیرند.»

محیط‌های ایمن برای مهندسی پرامپت

در برخی شرکت‌ها، کارگران دیجیتال جایگزین انسان‌ها شده‌اند، اما به گفته بیور، همچنان به تخصص توسعه‌دهندگان برای استفاده‌ی ایمن از این فناوری‌ها نیاز است. بیور مي گوید: «استفاده از ابزارهای ساخت Agentic و رابط‌های کاربری گرافیکی، سطح ورود را پایین آورده، اما افراد بدون دانش نرم‌افزار نباید به‌تنهایی از آن‌ها استفاده کنند. فرآیند تضمین کیفیت باید حتی سخت‌گیرانه‌تر باشد.»

LLMها ممکن است در صورت دریافت دسترسی API ناآگاهانه، اطلاعات تجاری را به‌صورت پیش‌بینی‌ناپذیر تغییر دهند. به همین دلیل، LinkedIn محیط‌های آزمایشی مهندسی پرامپت را با استفاده از Jupyter Notebook ایجاد کرده است.

رامگوپال می‌گوید: «در محصولات عامل‌محور، فقط محدودیت‌ها را تعریف می‌کنید. این محیط آزمایشی فرصتی است برای مدیران محصول و دانشمندان داده تا تجربه‌ی محصول را بدون محیط توسعه کامل بررسی کنند.»