دانشکده مهندسی کامپیوتر

بررسی مدل‌های زبانی بزرگ عامل‌محور در زمینه جستجوی معماری شبکه های عصبی و بهینه سازی ابرپارامتر

گزارش سمینار کارشناسی ارشد

در رشته مهندسی کامپیوتر گرایش هوش مصنوعی و رباتیکز

محمدصادق پولائی موزیرجی

اساتید راهنما

دکتر محمدرضا محمدی و دکتر سید صالح اعتمادی

آبان ماه ۱۴۰۴


بسم الله الرحمن الرحیم

چکیده

رویکردهای سنتی یادگیری ماشین خودکار با استفاده از بهینه‌سازی بیزی و الگوریتم‌های تکاملی فاقد تفسیرپذیری بوده و در بهره‌برداری از دانش حوزه‌ای دچار چالش هستند. مدل‌های زبانی بزرگ با توانایی‌های درک زبان طبیعی، استدلال و تولید کد، تحولی بنیادین ارائه می‌دهند. این سمینار بررسی می‌کند چگونه عامل‌های مبتنی بر مدل‌های زبانی بزرگ می‌توانند سیستم‌های یادگیری ماشین خودکار را با عمل به عنوان عامل‌های هوشمند، آگاه از زمینه و تطبیق‌پذیر دگرگون سازند. این پژوهش یک مرور نظام‌مند از رویکردهای اخیر از سه منظر ارائه می‌دهد: (۱) معماری عامل: سامانه‌های تک‌عاملی (بهینه‌سازی مستقیم، عملگرهای تکاملی، کنترل‌گرهای جریان) در مقابل چندعاملی (همکاری مبتنی بر نقش، هماهنگی سلسله‌مراتبی) (۲) منابع دانش: دانش درونی (تاریخچه بهینه‌سازی، یادگیری درون‌متنی) در برابر بازیابی بیرونی (استخراج از ادبیات، مخازن مدل) و (۳) قالب خروجی: پیکربندی‌های ساختاریافته، تولید کد، نمایش‌های درختی و رویکردهای ترکیبی. تحلیل نشان می‌دهد حوزه به طور مساوی بین معماری‌های تک‌عاملی و چندعاملی تقسیم شده، با تکیه اکثر سامانه‌ها بر دانش درونی و بهره‌گیری اندک از دانش بیرونی. قالب‌های خروجی تنوع بالایی دارند و رویکردهای ترکیبی تعادل بین خوانایی ماشینی و بیان‌پذیری را فراهم می‌کنند. سمینار مسائل باز و جهت‌های آینده شامل بهینه‌سازی با منابع محدود، یکپارچه‌سازی دانش پیشرفته و چارچوب‌های چندعاملی را شناسایی می‌کند. پیشنهاد پایان‌نامه برای طراحی سیستم عامل‌ـ‌محور جهت انتخاب و تنظیم مدل در انتقال یادگیری با داده محدود ارائه می‌شود.

واژگان کلیدی: یادگیری ماشین خودکار، مدل‌های زبانی بزرگ، سیستم‌های عامل‌ـ‌محور، بهینه‌سازی ابرپارامترها، جستجوی معماری عصبی، تولید تقویت‌شده با بازیابی، یادگیری درون‌متنی، انتقال یادگیری

فهرست مطالب

فهرست تصاویر

فهرست جداول

فصل ۱

مقدمه

۱-۱ شرح مسأله

فرایند ساخت و بهینه‌سازی مدل‌های یادگیری ماشین به‌طور سنتی، فرایندی پیچیده، زمان‌بر و نیازمند دانش تخصصی عمیق است. متخصصان علم داده ناچارند زمان قابل توجهی را صرف مهندسی ویژگی[۱]، انتخاب مدل[۲]، و تنظیم ابرپارامترها[۳] کنند. یادگیری ماشین خودکار[۴] با هدف خودکارسازی این خط لوله[۵] و مردمی‌سازی[۶] یادگیری ماشین پدیدار شد. با این حال، روش‌های سنتی یادگیری ماشین خودکار، مانند بهینه‌سازی بیزی[۷] یا الگوریتم‌های تکاملی[۸]، اغلب به عنوان بهینه‌سازهای جعبه‌سیاه[۹] عمل می‌کنند که فاقد تفسیرپذیری بوده و در انطباق با مسائل جدید یا بهره‌برداری از دانش برون‌حوزه‌ای دچار چالش هستند.

در سال‌های اخیر، ظهور مدل‌های زبانی بزرگ[۱۰] با توانایی‌های چشمگیر در درک زبان

[1] feature engineering
[2] model selection
[3] hyperparameter tuning
[4] Automated Machine Learning (AutoML)
[5] pipeline
[6] democratization
[7] Bayesian optimization
[8] evolutionary algorithms
[9] black-box
[10] Large Language Models (LLMs)

طبیعی[11]، استدلال و تولید کد، روش جدیدی را معرفی کرده است. این مدل‌ها پتانسیل آن را دارند که از بهینه‌سازهای کور فراتر رفته و به عنوان عامل[12]های هوشمند عمل کنند. این عامل‌ها می‌توانند با استفاده از دانش پیشین خود، استدلال گام‌به‌گام، و حتی تعامل با محیط‌های اجرایی و پایگاه‌های دانش[13]، فرایند یادگیری ماشین خودکار را به شیوه‌ای خودمختار[14]، تطبیق‌پذیر[15] و آگاه از زمینه[16] هدایت کنند. مسئله اصلی که این سمینار به آن می‌پردازد، بررسی این تقاطع نوظهور است: چگونه می‌توان از مدل‌های زبانی بزرگ عامل‌ـ محور برای دگرگونی و ارتقای نسل بعدی سیستم‌های یادگیری ماشین خودکار بهره جست؟

۱-۲ معرفی حوزه سمینار

حوزه اصلی این سمینار، یادگیری ماشین خودکار است. این حوزه به طور سنتی بر توسعه روش‌هایی برای خودکارسازی کامل خط لوله یادگیری ماشین، به‌ویژه چالش‌های محاسباتی پیچیده‌ای چون بهینه‌سازی ابرپارامترها[17] و جستجوی معماری عصبی[18]، تمرکز داشته است.

این سمینار یک روش نوظهور و دگرگون‌ساز در این حوزه را بررسی می‌کند که مبتنی بر ادغام مدل‌های زبانی بزرگ است. برخلاف روش‌های کلاسیک یادگیری ماشین خودکار (مانند بهینه‌سازی بیزی یا الگوریتم‌های تکاملی) که اغلب به عنوان بهینه‌سازهای جعبه‌سیاه عمل می‌کنند، مدل‌های زبانی بزرگ با توانایی‌های بی‌نظیر خود در درک دستورالعمل‌های پیچیده، استدلال گام‌به‌گام و تولید کد، پتانسیل ایجاد فرایندهای بهینه‌سازی شفاف‌تر، انعطاف‌پذیرتر و آگاهانه‌تر را فراهم می‌کنند.

نکته کلیدی که این سمینار به آن می‌پردازد، بررسی این مدل‌های زبانی در چارچوب سیستم‌های عامل‌ـ محور[19] است. در این دیدگاه، مدل زبانی به عنوان مغز متفکر یک عامل

[11] Natural Language Understanding (NLU)
[12] agent
[13] knowledge bases
[14] autonomous
[15] adaptive
[16] context-aware
[17] Hyperparameter Optimization (HPO)
[18] Neural Architecture Search (NAS)
[19] Agent-based Systems

هوشمند عمل می‌کند. این عامل می‌تواند به صورت خودمختار، وظایف پیچیده یادگیری ماشین خودکار را مدیریت کند، از دانش خارجی بهره ببرد، استراتژی‌های جستجو را استدلال نماید، کد اجرایی تولید کند و بر اساس بازخورد، رویکرد خود را تطبیق دهد.

بنابراین، حوزه تخصصی این سمینار، نقطه تلاقی این مفاهیم مطالعه و تحلیل کاربرد عامل‌های هوشمند مبتنی بر مدل زبانی بزرگ به منظور ارتقا و تحول در نسل جدید سیستم‌های یادگیری ماشین خودکار می‌باشد.

۱-۳ ساختار گزارش

این گزارش در ۴ فصل تهیه شده است. در فصل اول به بیان مقدمه و شرح مسئله پرداخته ایم. در فصل دوم مبانی و مفاهیم حوزه یادگیری ماشین خودکار، بهینه‌سازی ابرپارامترها و مدل های زبانی بزرگ عامل محور شرح داده شده است. در فصل سوم روش های مختلف از دیدگاه های عامل، دانش و نوع خروجی بررسی و مقایسه شده است. در فصل چهارم نیز به جمع‌بندی، نتیجه‌گیری از مطالب آورده شده در گزارش و کارهای آینده پرداخته‌ایم.

فصل ۲

تعاریف و مفاهیم مبنایی

۲–۱ مقدمه

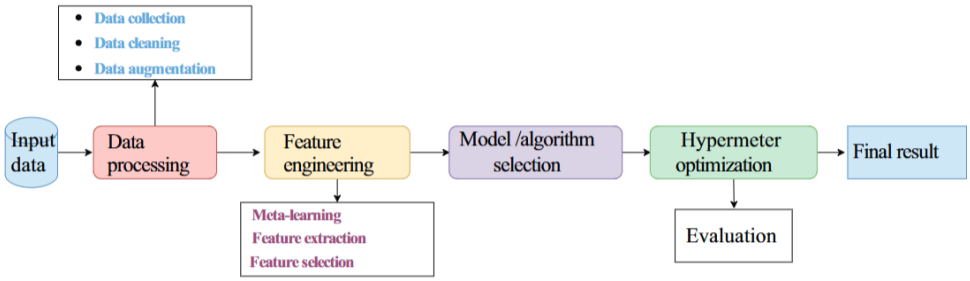
برای درک عمیق روش‌های نوین ارائه شده در این گزارش، ضروری است که با مفاهیم و تعاریف پایه‌ای در حوزه‌های یادگیری ماشین خودکار، مدل‌های زبانی بزرگ و سیستم‌های عامل‌ـ محور آشنا شویم. این فصل به مرور این مبانی نظری اختصاص دارد و به عنوان سنگ بنایی برای تحلیل‌های ارائه شده در فصل سوم عمل خواهد کرد.

۲–۲ یادگیری خودکار ماشین

یادگیری ماشین خودکار به فرایند خودکارسازی وظایف تکراری و تخصصی در طراحی و استقرار مدل‌های یادگیری ماشین اطلاق می‌شود. هدف نهایی یادگیری ماشین خودکار، کاهش نیاز به دخالت متخصصان انسانی و تسریع فرایند کشف و اعتبارسنجی مدل‌های کارآمد است. این فرایند معمولاً شامل مراحلی چون پیش‌پردازش داده[1]، مهندسی ویژگی، انتخاب مدل و بهینه‌سازی ابرپارامتر می‌باشد. همانطور که در شکل ۲–۱ نشان داده شده است، چارچوب یادگیری ماشین خودکار شامل مراحل کلیدی از پیش‌پردازش داده تا استقرار

[1] data preprocessing

مدل است. هر مرحله می‌تواند با استفاده از الگوریتم‌ها و تکنیک‌های مختلف خودکارسازی شود تا بهینه‌ترین مدل برای یک مسئله خاص یادگیری ماشین شناسایی گردد. مسئله **CASH**:



شکل ۲-۱: چارچوب یادگیری ماشین خودکار [۱].

صورت‌بندی و تفسیر یکپارچه. در یادگیری ماشین خودکار، مسئلهٔ CASH[۲] به‌دنبال یافتن زوج (A^*, λ^*) است که با انتخاب هم‌زمان نوع الگوریتم و پیکربندی ابرپارامترهای آن، کمترین خطا را روی داده‌های اعتبارسنجی (مثلاً در اعتبارسنجی متقابل kـ بخشی) حاصل کند [؟]:

$$A^*, \lambda^* \in \underset{A^{(j)} \in \mathcal{A},\, \lambda \in \Lambda^{(j)}}{\arg\min} \frac{1}{k} \sum_{i=1}^{k} \mathcal{L}\left(A_\lambda^{(j)},\, D_{\text{train}}^{(i)},\, D_{\text{valid}}^{(i)}\right). \qquad (۱-۲)$$

که در آن $\mathcal{A}$ مجموعهٔ الگوریتم‌های کاندید است و برای هر $A^{(j)} \in \mathcal{A}$ فضای ابرپارامتر $\Lambda^{(j)}$ تعریف می‌شود؛ $A_\lambda^{(j)}$ نمونه‌سازی الگوریتم j با پیکربندی $\lambda \in \Lambda^{(j)}$ است؛ $D_{\text{train}}^{(i)}$ و $D_{\text{valid}}^{(i)}$ به‌ترتیب داده‌های آموزش و اعتبارسنجی در تکرار i از اعتبارسنجی متقابل kـ بخشی‌اند؛ k تعداد بخش‌هایی است که خطا روی آن‌ها میانگین‌گیری می‌شود؛ و $\mathcal{L}(\cdot)$ تابع زیانی است که باید کمینه شود (برای معیارهای حداکثری مانند دقت یا AUC می‌توان از منفیِ امتیاز به‌عنوان زیان استفاده کرد). بدین‌ترتیب، جست‌وجو در CASH هم‌زمان روی فضای انتخاب الگوریتم و فضای ابرپارامترهای هر الگوریتم انجام می‌شود تا زوج بهینه (A^*, λ^*) با بهترین عملکرد اعتبارسنجی به‌دست آید.

[2] Combined Algorithm Selection and Hyperparameter Optimization

۲-۲-۱ بهینه‌سازی ابرپارامتر

ابرپارامترها[3] پارامترهایی در مدل یادگیری ماشین هستند که مقدار آن‌ها پیش از آغاز فرایند آموزش تنظیم می‌شود (برخلاف پارامترها یا وزن‌ها که در طول آموزش یادگرفته می‌شوند). بهینه‌سازی ابرپارامتر فرایند یافتن ترکیبی از ابرپارامترها است که منجر به بهترین کارایی[4] مدل بر روی یک مجموعه داده اعتبارسنجی می‌شود. روش‌های متداول برای این کار شامل جستجوی شبکه‌ای[5]، جستجوی تصادفی[6] و بهینه‌سازی بیزی است. این فرایند به دلیل هزینه محاسباتی[7] بالای ارزیابی هر پیکربندی[8]، بسیار چالش‌برانگیز است.

$$\lambda^\star = \arg\min_{\lambda} \ \mathcal{L}_V^\star(\lambda) = \arg\min_{\lambda} \ \mathcal{L}_V\big(\lambda, \mathbf{w}^\star(\lambda)\big), \tag{۲-۲}$$

$$\text{s.t.} \quad \mathbf{w}^\star(\lambda) = \arg\min_{\mathbf{w}} \ \mathcal{L}_T(\lambda, \mathbf{w}). \tag{۳-۲}$$

در معادله (۲-۲)، هدف کمینه‌کردن زیان اعتبارسنجی با پارامترهای پاسخ بهینه $\mathcal{L}_V^\star$ است؛ این کمینه‌سازی زمانی معنا دارد که معادله (۳-۲) تا همگرایی حل شده باشد. در این فرمول‌ها، $\mathcal{L}_T$ و $\mathcal{L}_V$ به‌ترتیب اهداف آموزش و اعتبارسنجی هستند و λ و $\mathbf{w}$ نیز به‌ترتیب ابرپارامترها و پارامترهای مدل‌اند. بهینه‌سازی ابرپارامتر می‌تواند به‌صورت ترتیبی انجام شود؛ بدین معنا که پیشنهاد λ_n به دنباله مقادیر قبلی $\{\lambda_1, \lambda_2, \ldots, \lambda_{n-1}\}$ و زیان‌های اعتبارسنجی متناظرشان وابسته است.

[3] Hyperparameters
[4] performance
[5] Grid Search
[6] Random Search
[7] computational cost
[8] configuration

۲-۲-۲ جستجوی معماری شبکه عصبی

جستجوی معماری عصبی یکی از زیرشاخه‌های یادگیری ماشین خودکار است که به طور خاص بر خودکارسازی طراحی معماری شبکه‌های عصبی عمیق[۹] تمرکز دارد. به جای تکیه بر شهود[۱۰] و طراحی دستی توسط متخصصان، جستجوی معماری عصبی از الگوریتم‌های جستجو[۱۱] (مانند یادگیری تقویتی[۱۲] یا روش‌های تکاملی[۱۳]) برای کاوش در فضای طراحی[۱۴] وسیع معماری‌ها استفاده می‌کند. هدف، یافتن معماری‌ای است که بهترین توازن را میان دقت و کارایی محاسباتی[۱۵] برقرار کند.

۲-۳ مدل های زبانی بزرگ

مدل‌های زبانی بزرگ مدل‌های زبانی هستند که با استفاده از معماری ترنسفورمر و بر روی حجم عظیمی از داده‌های متنی آموزش دیده‌اند. این مدل‌ها دارای میلیاردها پارامتر هستند و توانایی‌های شگفت‌انگیزی در درک زمینه[۱۶]، تولید متن منسجم[۱۷] و استدلال از خود نشان می‌دهند.

۲-۳-۱ تاریخچه

توسعه مدل‌های زبانی بزرگ با معرفی معماری ترنسفورمر [۲] شتاب گرفت. مدل‌هایی مانند BERT [۳] و GPT [۴] چارچوب نظری[۱۸] پیش‌آموزش[۱۹] و ریزتنظیم[۲۰] را تثبیت کردند.

[9] Deep Neural Networks
[10] intuition
[11] search algorithms
[12] Reinforcement Learning
[13] evolutionary methods
[14] design space
[15] computational efficiency
[16] understanding context
[17] coherent
[18] paradigm
[19] pre-training
[20] fine-tuning

نسل‌های بعدی این مدل‌ها، با افزایش چشمگیر مقیاس[۲۱] داده و پارامترها، به توانایی‌های یادگیری زمینه‌ای[۲۲] دست یافتند که به آن‌ها اجازه می‌دهد وظایف جدید را بدون ریزتنظیم و تنها با دیدن چند مثال در دستور[۲۳] انجام دهند.

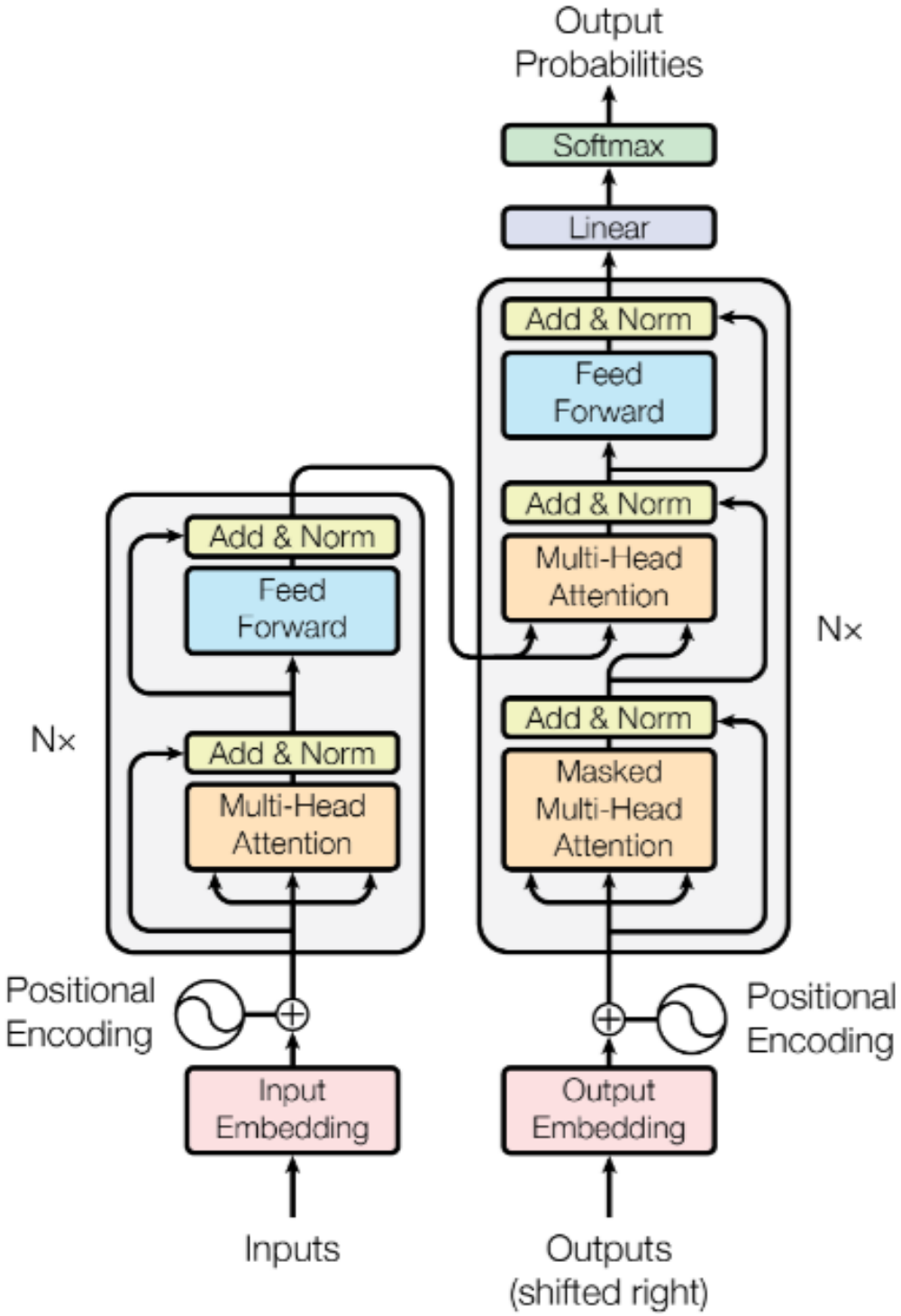
۲-۳-۲ معماری

پایه و اساس اکثر مدل‌های زبانی بزرگ مدرن، معماری ترنسفورمر است که بر مکانیزم توجه خودی[۲۴] تکیه دارد. این مکانیزم به مدل اجازه می‌دهد تا وابستگی‌های دوربرد[۲۵] را در متن مدل کند و به بخش‌های مختلف ورودی وزن‌های متفاوتی اختصاص دهد. همانطور که در شکل ۲-۲ نشان داده شده است، مدل‌ها معمولاً از پشته‌ای از لایه‌های رمزگذار[۲۶] (مانند BERT) یا رمزگشا[۲۷] (مانند GPT) یا هر دو (مانند T5) تشکیل شده‌اند [۲].

۲-۳-۳ کاربردها

مدل‌های زبانی بزرگ کاربردهای متنوعی از جمله ترجمه ماشینی[۲۸] [۵]، خلاصه‌سازی متن[۲۹]، پرسش و پاسخ و تولید محتوا دارند. اخیراً، توانایی آن‌ها در تولید کد[۳۰] [۶] و حل مسائل منطقی [۷]، درهای جدیدی را برای استفاده از آن‌ها در حوزه‌های فنی‌تر مانند مهندسی نرم‌افزار و یادگیری ماشین خودکار گشوده است.

[21] scale
[22] In-Context Learning (ICL)
[23] prompt
[24] self-attention
[25] long-range dependencies
[26] encoder
[27] decoder
[28] machine translation
[29] text summarization
[30] code generation



شکل ۲-۲: معماری ترنسفورمر [۲].

۲-۳-۴ تفکر و عمل

فراتر از تولید پاسخ‌های ایستا[۳۱]، مدل‌های زبانی بزرگ می‌توانند برای تفکر[۳۲] و عمل[۳۳] نیز به کار روند. چارچوب‌هایی مانند ReAct [۸] نشان دادند که چگونه یک مدل زبانی بزرگ می‌تواند به صورت درهم‌تنیده[۳۴]، ردپاهای استدلالی[۳۵] (تفکر) و اقدامات (مانند جستجو در وب یا اجرای دستور) تولید کند. این قابلیت، سنگ بنای استفاده از مدل‌های زبانی بزرگ به عنوان هسته تصمیم‌گیرنده در عامل‌های خودمختار است. شکل ۲-۳ نمونه‌ای از این چارچوب را نشان می‌دهد.

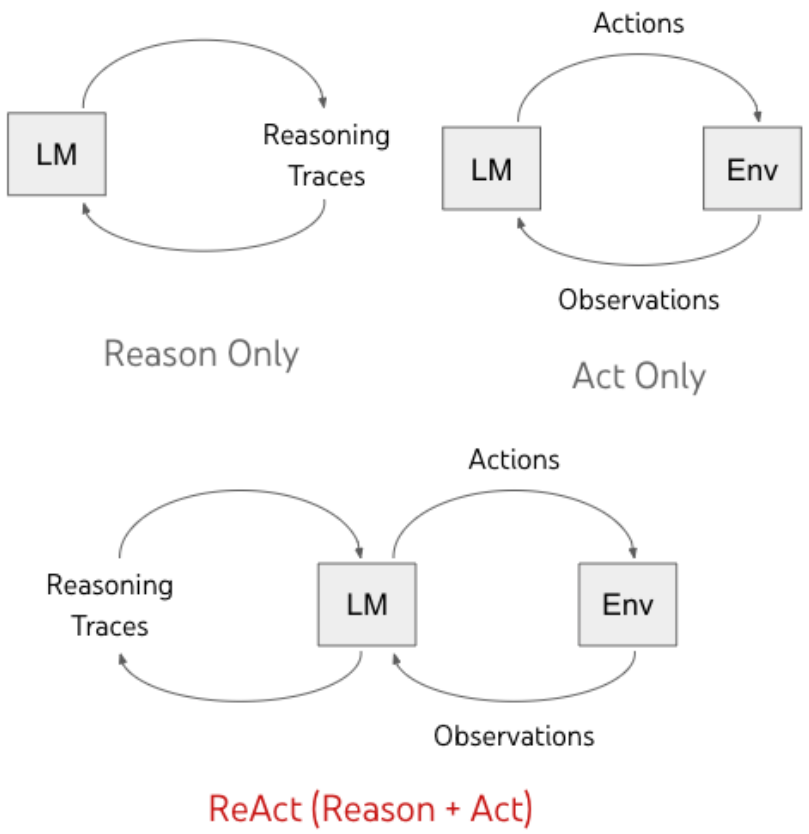
۲-۴ عامل

در زمینه هوش مصنوعی، عامل به سیستمی اطلاق می‌شود که در یک محیط[۳۶] قرار دارد، آن را از طریق حسگرها[۳۷] ادراک می‌کند و از طریق کنشگرها[۳۸] بر آن تأثیر می‌گذارد تا به اهداف خود دست یابد. عامل‌های زبانی نوع خاصی از عامل‌ها هستند که از مدل‌های زبانی بزرگ به عنوان موتور استدلال اصلی خود برای پردازش ادراکات (اغلب متنی)، برنامه‌ریزی[۳۹] و انتخاب اقدام استفاده می‌کنند [۹].

۲-۴-۱ تک‌عاملی[۴۰]

یک سیستم شامل یک عامل واحد است که تمام وظایف ادراک، استدلال و عمل را به تنهایی انجام می‌دهد. در زمینه یادگیری ماشین خودکار، این می‌تواند یک عامل مبتنی بر مدل زبانی بزرگ باشد که کل خط لوله بهینه‌سازی را از ابتدا تا انتها مدیریت می‌کند [۹].

[31] static responses
[32] Reasoning
[33] Action
[34] interleaved
[35] reasoning traces
[36] environment
[37] sensors
[38] actuators
[39] planning
[40] Single-Agent



شکل ۲-۳: چارچوب ReAct [۸].

۲-۴-۲ چندعاملی[۴۱]

سیستم‌های چندعاملی شامل دو یا چند عامل هستند که در یک محیط مشترک با یکدیگر تعامل[۴۲] می‌کنند. این تعامل می‌تواند همکارانه[۴۳] (برای دستیابی به یک هدف مشترک)، رقابتی[۴۴]، یا ترکیبی از هر دو باشد. در مسائل پیچیده، بهره‌گیری از چند عامل امکان تفکیک وظایف[۴۵]، موازی‌سازی، و افزایش پایداری در برابر خطا را فراهم می‌کند. برای

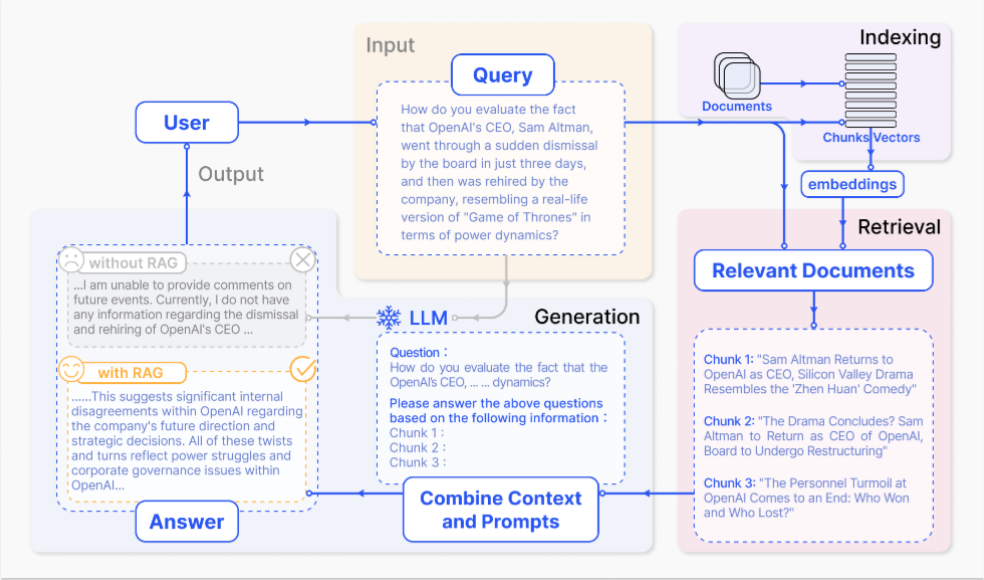
[41] Multi-Agent
[42] interact
[43] collaborative
[44] competitive
[45] task decomposition

نمونه، یک عامل متخصص تحلیل داده، یک عامل مسئول برنامه‌ریزی/تولید راه‌حل[۴۶]، و یک عامل منتقد[۴۷] وظیفه ارزیابی خروجی و ارائه بازخورد را بر عهده می‌گیرند؛ گاه یک عامل هماهنگ‌کننده[۴۸] نیز برای زمان‌بندی و حل تعارض‌ها [۴۹] به کار می‌رود. هماهنگی میان عامل‌ها می‌تواند متمرکز یا غیرمتمرکز باشد و معمولاً از طریق پیام‌رسانی، اشتراک حافظه، یا پروتکل‌های ارتباطی[۵۰] انجام می‌شود. از چالش‌های کلیدی می‌توان به تعریف اهداف/پاداش‌ها، تخصیص منابع، و مدیریت ناهمگونی مهارت‌ها اشاره کرد [۹].

۲-۵ تولید تقویت‌شده با بازیابی[۵۱]

تولید تقویت‌شده با بازیابی [۱۰] روشی است که مدل‌های زبانی بزرگ را با یک سازوکار بازیابی اطلاعات[۵۲] خارجی ترکیب می‌کند. به جای تکیه صرف بر دانش پارامتری (ذخیره شده در وزن‌های مدل)، تولید تقویت‌شده با بازیابی ابتدا اطلاعات مرتبط را از یک مجموعه متون[۵۳] یا پایگاه دانش[۵۴] بازیابی می‌کند و سپس این اطلاعات را به مدل زبانی بزرگ می‌دهد تا پاسخ نهایی را بر اساس آن تولید کند. این روش به کاهش توهم[۵۵] و افزایش دقت و به‌روز بودن[۵۶] اطلاعات کمک می‌کند [۱۱].در شکل ۲-۴، چارچوب کلی تولید تقویت‌شده با بازیابی نشان داده شده است.

[46] planning/solution generation
[47] critic
[48] coordinator
[49] conflict resolution
[50] communication protocols
[51] Retrieval-Augmented Generation (RAG)
[52] information retrieval mechanism
[53] corpus
[54] knowledge base
[55] hallucination
[56] up-to-dateness



شکل ۲-۴: چارچوب کلی تولید تقویت‌شده با بازیابی. این فرآیند شامل سه مرحله اصلی است: ۱) نمایه‌سازی، که در آن اسناد به قطعات برداری تبدیل و ذخیره می‌شوند؛ ۲) بازیابی، که در آن مرتبط‌ترین قطعات بر اساس شباهت معنایی با پرسش استخراج می‌شوند؛ و ۳) تولید، که در آن مدل زبانی بزرگ با استفاده از پرسش و قطعات بازیابی‌شده، پاسخ نهایی را تولید می‌کند [۱۲].

۲-۵-۱ پایگاه دانش

پایگاه دانش در تولید تقویت‌شده با بازیابی معمولاً مجموعه‌ای از اسناد[۵۷] است. این اسناد اغلب به قطعات[۵۸] کوچکتر تقسیم شده و به صورت نمایش‌های برداری[۵۹] (یا نهفتگی‌ها[۶۰]) در یک پایگاه داده برداری[۶۱] ذخیره می‌شوند تا بازیابی مبتنی بر شباهت معنایی[۶۲] به سرعت انجام شود.

[57] documents
[58] chunks
[59] vector representations
[60] embeddings
[61] vector database
[62] semantic similarity retrieval

۲-۵-۲ ترکیب با عامل

عامل‌های هوشمند می‌توانند از تولید تقویت‌شده با بازیابی به عنوان یک ابزار[۶۳] کلیدی استفاده کنند. زمانی که یک عامل یادگیری ماشین خودکار با یک مجموعه داده جدید روبرو می‌شود، می‌تواند از تولید تقویت‌شده با بازیابی برای جستجو در پایگاه دانش استفاده کند. این دانش بازیابی‌شده به عامل کمک می‌کند تا تصمیمات آگاهانه‌تری در مورد مسئله اتخاذ کند [۱۳].

[63] tool

فصل ۳

مروری بر کارهای مرتبط

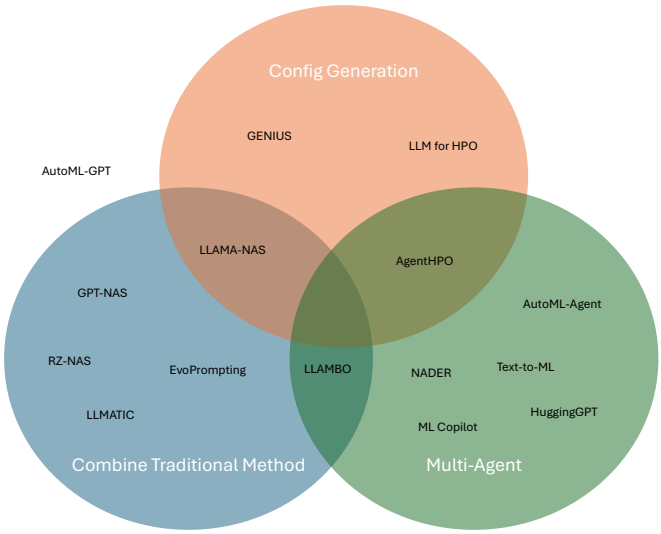
۳-۱ مقدمه

ادغام مدل‌های زبانی بزرگ در خودکارسازی یادگیری ماشین به‌عنوان رویکردی دگرگون‌ساز برای بهینه‌سازی معماری‌های عصبی و فراپارامترها پدیدار شده است. این فصل کارهای موجود را از سه منظر مکمل بررسی می‌کند: معماری عامل، راهبردهای تقویت دانش[1]، و طراحی قالب خروجی. این ابعاد در کنار هم نشان می‌دهند که سامانه‌های گوناگون چگونه از قابلیت‌های مدل‌های زبانی بزرگ بهره می‌برند و هم‌زمان محدودیت‌های ذاتی آن‌ها را مدیریت می‌کنند. در شکل ۳-۱، دسته‌بندی مقالات مرورشده بر مبنای این سه منظر ارائه شده است. بخش‌های بعدی به‌تفصیل هر یک از این ابعاد را بررسی می‌کنند و نمونه‌های شاخص را برای روشن‌ساختن رویکردهای مختلف ارائه می‌دهند.

۳-۲ ظهور نخستین روش‌های یادگیری ماشین خودکار

خودکارسازی یادگیری ماشین با هدف مردمی‌سازی فرایندهای پرهزینه و زمان‌بر گزینش مدل و بهینه‌سازی فراپارامتر پدید آمد. ریشه گزینش الگوریتم به سال ۱۹۷۶ بازمی‌گردد

[1] knowledge augmentation strategies



شکل ۳-۱: دسته بندی مقالات بر مبنای معماری عامل، استفاده از روش های سنتی، و قالب خروجی مدل زبانی بزرگ.

که در آن، مسئله انتخاب الگوریتم بهینه از میان یک مجموعه ازپیش‌تعریف‌شده به‌منظور کمینه‌سازی افت کارایی روی داده‌های اعتبارسنجی صورت‌بندی شد [۱۴]. کوشش‌های نخستین در حوزه بهینه‌سازی فراپارامتر بر روش‌های منفردی چون جست‌وجوی شبکه‌ای و جست‌وجوی تصادفی متمرکز بود؛ روش‌هایی ساده اما ناکارا در فضاهای بعدبالا [۱۵]. این روش نشان داد که جست‌وجوی تصادفی با تخصیص مؤثرتر منابع به فراپارامترهای اثرگذار، از جست‌وجوی شبکه‌ای پیشی می‌گیرد و بدین‌ترتیب زمینه را برای روش‌های پیشرفته‌تر فراهم ساخت [۱۵].

نقطه عطف در ۲۰۱۳ با معرفی مسئله CASH رخ داد؛ چارچوبی که گزینش الگوریتم و بهینه‌سازی فراپارامتر را در یک بهینه‌سازی واحد ادغام می‌کرد [۱۶]. بر مبنای این ایده، Auto-WEKA به‌عنوان نخستین سامانه جامع یادگیری ماشین خودکار پدید آمد و با اتکا به SMAC رویکردی مبتنی بر بهینه‌سازی بیزی که از جنگل‌های تصادفی[۲] به‌عنوان جانشین[۳] استفاده می‌کرد به جست‌وجو در میان رده‌بندهای WEKA و فراپارامترهایشان پرداخت [۱۶، ۱۷]. در ادامه، auto-sklearn در ۲۰۱۵ روش CASH را به الگوریتم‌های scikit-

[2] random forests
[3] surrogate

learn بسط داد و با به‌کارگیری فرا-یادگیری[۴] برای شروع تازه و نیز ساخت هم‌بندی[۵]های مقاوم، بهبودهای معناداری رقم زد [۱۸].

به‌طور موازی، الگوریتم‌های تکاملی به‌عنوان بدیلی تواناتر مطرح شدند؛ چنان‌که TPOT خطوط لوله یادگیری ماشین را به‌صورت برنامه‌هایی درخت‌ساختار مدل می‌کند و با برنامه‌نویسی ژنتیکی[۶]، ترکیب‌های انعطاف‌پذیری از پیش‌پردازنده‌ها، گزیننده‌های ویژگی[۷] و مدل‌ها را می‌آزماید [۱۹]. روش‌های چندسطوحی/چندوفایی[۸] با تخصیص تدریجی منابع به پیکربندی‌های امیدبخش و با الهام از راهبردهای باندیتی[۹] هزینه محاسباتی را مهار کردند [۲۰، ۲۱]. تمرکز این رویکردهای اولیه عمدتاً بر مسائل کلاسیک طبقه‌بندی[۱۰] و رگرسیون[۱۱] بود؛ و معیارهایی چون OpenML-CC18 ارزیابی‌های استانداردشده را تسهیل کردند [۲۲].

گذار به جست‌وجوی معماری شبکه‌های عصبی از حوالی ۲۰۱۷ آغاز شد و طراحی معماری را امتداد طبیعی بهینه‌سازی فراپارامتر تلقی کرد. کارهای اولیه مبتنی بر یادگیری تقویتی در جست‌وجوی معماری شبکه‌های عصبی، گرچه پرهزینه، نقطه شروعی اثرگذار بودند و گونه‌های کاراتری همچون ENAS با اشتراک پارامتر[۱۲] را الهام بخشیدند [۲۳، ۲۴]. در ادامه، معیارهایی مانند NAS-Bench-101 با فراهم‌کردن ارزیابی‌های ازپیش‌محاسبه‌شده، امکان پژوهش بازتولیدپذیر را مهیا کردند [۲۵]. این سیر تحول، تنظیم دستی را به خطوط لوله خودکار بدل کرد و بستر را برای همگرایی با روش‌های پیشرفته‌تر فراهم ساخت.

[4] meta-learning
[5] ensemble
[6] genetic programming
[7] feature selectors
[8] multi-fidelity
[9] bandit-based strategies
[10] classification
[11] regression
[12] parameter sharing

۳-۳ تحلیل بر مبنای معماری عامل

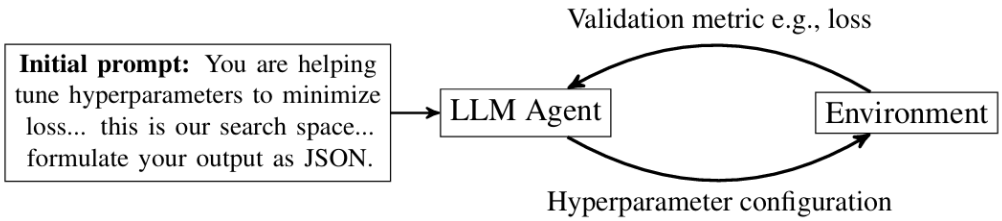
۳-۳-۱ سامانه‌های تک‌عاملی

بیشینه سامانه‌های مبتنی بر مدل‌های زبانی بزرگ برای خودکارسازی یادگیری ماشین، معماری‌های تک‌عاملی را برمی‌گزینند که در آن یک عامل منفرد کل جریان بهینه‌سازی را مدیریت می‌کند. بر مبنای چگونگی ادغام مدل زبانی در فرایند بهینه‌سازی، این سامانه‌ها در چند روش عملیاتی متمایز قرار می‌گیرند.

بهینه‌سازی مستقیم از طریق دستوردهی تکراری[۱۳]

در این رویکرد برجسته، مدل‌های زبانی به‌منزله بهینه‌سازهای جعبه‌سیاه به‌کار می‌روند که پیکربندی‌ها را پیشنهاد می‌کنند و از راه حلقه‌های بازخورد[۱۴] آن‌ها را پالایش می‌کنند. مدل با تکیه بر تاریخچه آزمون‌ها[۱۵] که به‌صورت گفت‌وگوهای چت[۱۶] یا خلاصه‌های فشرده انباشته می‌شود، زمینه[۱۷] را حفظ می‌کند و پالایش تکراری مبتنی بر معیارهای اعتبارسنجی را ممکن می‌سازد (شکل ۳-۲) [۲۶، ۲۷]. این روش به چارچوب‌های بهینه‌سازی بیزی نیز بسط یافته است؛ جایی که مدل‌های آغاز گرم، نمونه‌گیری از نامزدها و مدل‌سازی جانشین[۱۸] را با اتکاء به استدلال زبان طبیعی و مشروط بر تاریخچه بهینه‌سازی انجام می‌دهند [۲۸]. این رویکردها در تنظیمات با بودجه کم کارایی رقابتی نشان می‌دهند و بی‌آن‌که به ریزتنظیم نیاز داشته باشند، بر یادگیری زمینه‌ای از توصیف مسئله و بازخورد تجربی تکیه می‌کنند.

[13] iterative prompting
[14] feedback loops
[15] trials
[16] chat-style dialogues
[17] context
[18] surrogate modeling



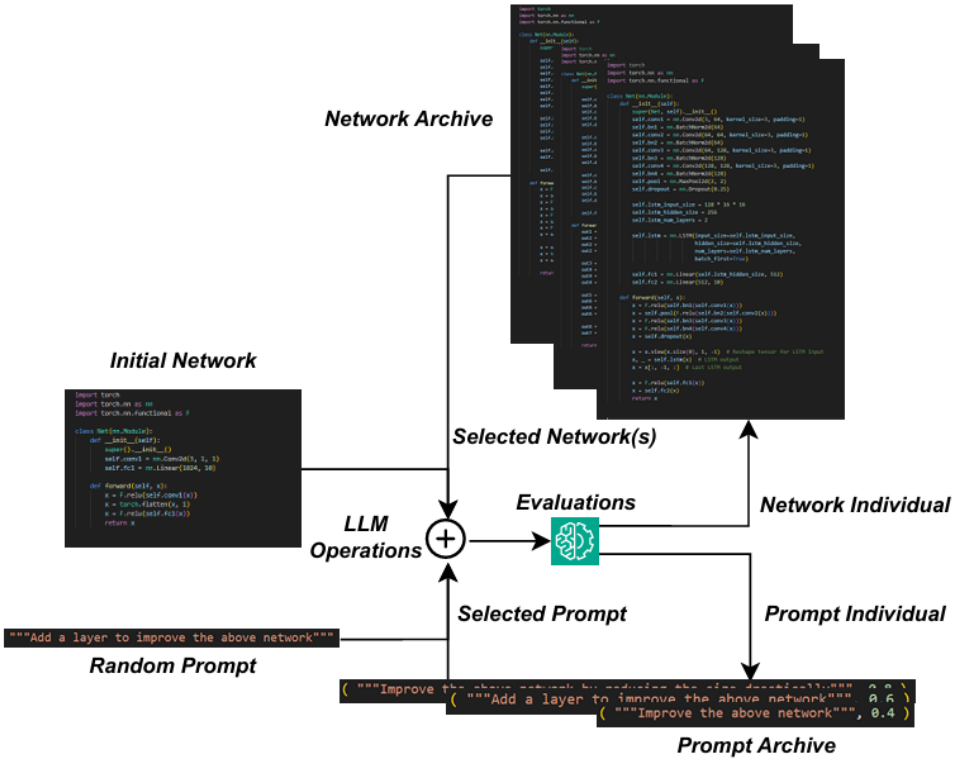
شکل ۳-۲: مدل‌های زبانی بزرگ برای بهینه‌سازی ابرپارامترها. در این چارچوب تصمیم‌گیری ترتیبی، توصیف مسئله و فضای جست‌وجو را به‌عنوان دستور به مدل زبانی داده می‌شود. سپس مدل زبانی مجموعه‌ای از ابرپارامترها را برای ارزیابی پیشنهاد می‌کند. محیط یک اجرای آموزش را با این تنظیم ابرپارامتر اجرا می‌کند و سپس مقدار یک معیار اعتبارسنجی دوباره به‌عنوان دستور به مدل زبانی داده می‌شود [۲۶].

عملگرهای تکاملی[۱۹]

راهبردی دیگر، مدل زبانی را به‌عنوان عملگرهای جهش[۲۰] یا ترکیب[۲۱] درون چارچوب‌های تکاملی جا می‌دهد. به‌جای جایگزینی الگوریتم‌های جست‌وجوی سنتی، این سامانه‌ها آن‌ها را با تنوع‌های تولیدشده توسط مدل زبانی تقویت می‌کنند. مدل می‌تواند تغییرات معماری مبتنی بر کد را در چارچوب‌های کیفیت—تنوع[۲۲] بسازد (شکل ۳-۳) [۲۹] یا به‌صورت عملگرهای تطبیقی که میان نسل‌ها با تنظیم دستور[۲۳] پالایش می‌شوند عمل کند [۳۰]. برخی پیاده‌سازی‌ها توانایی‌های بازتابی[۲۴] را نیز می‌گنجانند؛ به این معنا که مدل پیامدهای جهش را تحلیل می‌کند و بازخورد زبانی[۲۵] برای هدایت تکرارهای بعدی تولید می‌کند [۳۱]. این ادغام، پایداری[۲۶] جست‌وجوی تکاملی را حفظ می‌کند و در عین حال از خلاقیت مدل در تولید تنوع‌های معنادار بهره می‌گیرد. نمونه‌ای شاخص، GPT-NAS است که در آن مدل زبانی به‌منزله یک بازساز[۲۷] معماری عمل کرده و با ماسک‌گذاری و بازتولید لایه‌ها،

[19] Evolutionary Operators
[20] mutation
[21] crossover
[22] quality-diversity
[23] prompt-tuning
[24] reflective capabilities
[25] linguistic feedback
[26] robustness
[27] reconstructor

نامزدهای نمونه‌برداری‌شده توسط الگوریتم ژنتیک[۲۸] را بهبود می‌دهد؛ بنابراین عملاً نقشی هم‌ارز با یک عملگر جهش آگاه از زمینه ایفا می‌کند بی‌آن‌که راهبرد جست‌وجوی تکاملی را جایگزین کند [۳۲].



شکل ۳-۳: در این شکل، روند روش تکاملی نمایش داده شده است. در دور اولیه تکامل، یک شبکه اولیه با یک دستور تصادفی تحت یک عملیات جهش قرار می‌گیرد. سپس فردِ شبکه و فردِ دستور ارزیابی شده و در آرشیوهای جداگانه ذخیره می‌شوند. در طول حلقه تکاملی، دستور و شبکه انتخاب‌شده تحت یک عملیات تکاملی قرار می‌گیرند (در صورت استفاده از عملگر ترکیب، دستور ثابت باقی می‌ماند) تا شبکه‌ها و دستورهای بیشتری برای پر کردن و روشن‌سازی آرشیوها ایجاد شوند [۲۹].

[28] genetic algorithm

کنترل‌گرهای جریان کار

در این روش، مدل‌های زبانی به‌مثابه هماهنگ‌کننده[۲۹] برای مدیریت اجزای خط لوله به کار می‌روند. سامانه با ترکیب دستورهایی که فراداده ساختاریافته[۳۰] ــ از جمله کارت‌های داده[۳۱] و کارت‌های مدل[۳۲] ــ را در خود دارند، مدل را در مراحل پیاپی از پردازش داده، انتخاب مدل تا تنظیم فراپارامتر هدایت می‌کند [۳۳، ۳۴]. برخی پیاده‌سازی‌ها برنامه‌های پیچیده یادگیری ماشین را به مولفه‌های ماژولار[۳۳] تجزیه می‌کنند که به‌طور جداگانه تولید و با آزمون‌های واحد خودکار[۳۴] راستی‌آزمایی می‌شوند تا سازگاری تضمین گردد [۳۵]. این رویکرد با شکستن خطوط لوله طولانی و ناهمگون به زیروظایف قابل مدیریت و اتکاء به دستوردهی زمینه‌مند[۳۵]، انسجام کلی را حفظ می‌کند.

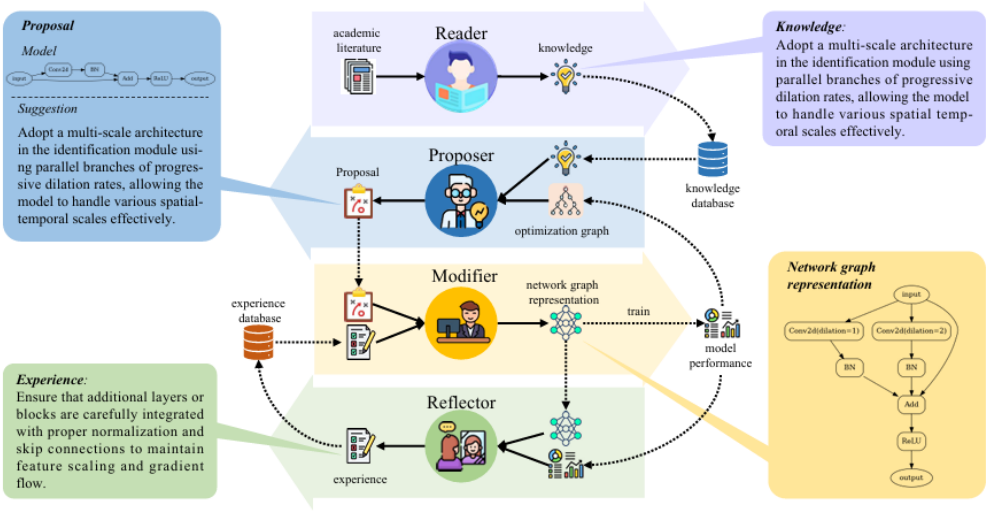
۳-۳-۲ سامانه‌های چندعاملی

معماری‌های چندعاملی با تفکیک نقش، زیروظایف یادگیری ماشین خودکار را میان عامل‌هایی با قابلیت‌های مکمل توزیع می‌کنند. این سامانه‌ها از رهگذر تفکیک کارکردی[۳۶] و همکاری بین‌عاملی[۳۷]، مدیریت پیچیدگی و استدلال پیچیده‌تر را ممکن می‌سازند.

همکاری مبتنی بر نقش[۳۸]

الگوی پایه، دو عامل تخصصی با مسئولیت‌های متمایز را به‌کار می‌گیرد. در یک ساختار، تولید پیکربندی از اجرای تجربی جدا می‌شود: عامل سازنده نیازمندی‌ها را تفسیر و پیکربندی‌های پیشنهادی همراه با استدلال ارائه می‌کند و عامل اجراکننده آموزش را انجام داده و نتایج را در

[29] orchestrator
[30] structured metadata
[31] data cards
[32] model cards
[33] modular components
[34] automated unit tests
[35] contextual prompting
[36] functional decomposition
[37] inter-agent collaboration
[38] Role-Based Collaboration



شکل ۳-۴: نمای کلی از چارچوب مبتنی بر نقش. خواننده به‌طور مداوم از ادبیات دانشگاهی می‌آموزد، در حالی که پیشنهاددهنده امیدوارکننده‌ترین شبکه‌های نامزد را شناسایی کرده و اصلاحاتی را پیشنهاد می‌کند. اصلاح‌کننده این پیشنهادها را پیاده‌سازی می‌کند و بازتاب‌دهنده نتایج را تحلیل و بازخورد ارائه می‌دهد. عملکرد شبکه اصلاح‌شده به پیشنهاددهنده بازگردانده می‌شود تا پیشنهادهای بعدی را آگاه سازد و یک چرخه بهبود مستمر را تقویت کند [۳۶].

گزارش‌های مشترک می‌گنجاند تا چرخه‌های بعدی پیشنهادهای سازنده را تغذیه کند [۳۷]. این تقسیم کار بازتاب گردش‌کار متخصصان است و با حافظه ترتیبی[۳۹] انباشته، به عملکرد خودگردان بدون مداخله انسانی می‌انجامد. افزون بر این، همین الگو را می‌توان به سطح تیمی تعمیم داد: نمونه پیشرفته، تقسیم عامل‌ها به تیم پژوهش[۴۰] و تیم توسعه[۴۱] است که به‌ترتیب دانش را از ادبیات پژوهشی[۴۲] استخراج و پیشنهادهای تغییر را می‌سازند، و آن پیشنهادها را بر نمایش‌های گراف[۴۳] اعمال کرده و هم بازخورد فوری و هم استخراج تجربه بلندمدت فراهم می‌کنند (شکل ۳-۴) [۳۶]. در این قالب، پایگاه‌های داده برداری[۴۴] با بازیابی مبتنی بر شباهت[۴۵] برای آمیختن دانش ادبیات با سوابق طراحی به‌کار می‌روند تا چرخه‌های بعدی

[39] episodic memory
[40] Research Team
[41] Development Team
[42] literature
[43] graph representations
[44] vector databases
[45] similarity-based retrieval

پیشنهاددهی و اجرا بهتر هدایت شوند.

هماهنگی سلسله‌مراتبی[۴۶]

در ساختارهای پیچیده‌تر، چند عامل تخصصی تحت نظارت یک مدیر عامل[۴۷] سازمان می‌یابند. مدیر با استدلال تقویت‌شده با بازیابی[۴۸] طرح‌های متنوعی می‌سازد، آن‌ها را به زیروظایف قابل موازی‌سازی[۴۹] واگشایی و به عامل مناسب تخصیص می‌دهد و از رهگذر راستی‌آزمایی چندمرحله‌ای و حلقه‌های بازنگری[۵۰] نتایج را اعتبارسنجی می‌کند [۳۸].

۳-۴ تحلیل منابع دانش[۵۱]

کارآمدی سامانه‌های خودکارسازی یادگیری ماشین مبتنی بر مدل‌های زبانی بزرگ به‌صورت بنیادین به چگونگی اکتساب، مدیریت و بهره‌برداری از دانش در سراسر فرایند بهینه‌سازی وابسته است. رویکردهای معاصر طیفی از راهبردهای تقویت دانش را پوشش می‌دهند؛ از یادگیری درون‌متنی صرف تا چارچوب‌های تولید تقویت‌شده با بازیابی که هر یک پیامدهای متمایزی برای کارایی جست‌وجو و تعمیم‌پذیری[۵۲] دارند.

۳-۴-۱ دانش درونی: تاریخچه آزمون و بازتاب[۵۳]

این رسته، دانش تولیدشده توسط خود سامانه را در بر می‌گیرد از اتکای مستقیم به تاریخچه آزمون‌ها در پنجره زمینه[۵۴] تا حافظه رویدادی و خودبازتابی[۵۵] که به‌تدریج به بازخورد قابل

[46] Hierarchical Coordination
[47] Agent Manager
[48] retrieval-augmented reasoning
[49] parallelizable subtasks
[50] revision loops
[51] Knowledge Source Analysis
[52] generalization
[53] System-Internal Knowledge: Trials and Reflection
[54] context window
[55] self-reflection

اقدام[۵۶] و اصول طراحی تقطیر می‌شوند.

```
Epoch:      [0, 5, 10, 15, 20]
Train Acc:  [39.54, 82.49, 96.30, 97.73, 98.30]
Val Acc:    [50.3, 77.76, 82.73, 81.34, 82.69]
Total Training Time: 1442.33s | Final Valid Acc: 82.43%

The model showed signs of overfitting, as indicated by the
perfect training accuracy and the plateauing validation
accuracy.
```

شکل ۳-۵: یادگیری درون‌متنی از تاریخچه بهینه‌سازی [۳۷].

یادگیری درون‌متنی از تاریخچه بهینه‌سازی[۵۷]

یک راهبرد رایج، پنجره زمینه مدل را همچون مخزن اصلی دانش در نظر می‌گیرد و صرفاً بر تاریخچه آزمون‌های انباشته‌شده طی فرایند بهینه‌سازی[۵۸] تکیه می‌کند. سامانه‌های مبتنی بر این ایده، پیکربندی‌های پیشین و سنجه‌های کارایی متناظر را در دستور می‌گنجانند تا مدل بتواند از رهگذر بازخورد تکرارشونده پیشنهادها را پالایش کند [۲۶، ۲۷، ۲۸]. این تاریخچه ممکن است به قالب‌های گوناگونی سریال‌سازی[۵۹] شود: گفت‌وگوهای سبک‌گپ[۶۰] که توالی زمانی تعاملات را حفظ می‌کنند، خلاصه‌های فشرده برای مهار قیود طول زمینه، الگوهای اندک‌نمونه[۶۱] برگزیده از جمعیت ارزیابی‌شده [۲۶، ۳۰]، و نیز تجربه‌های تاریخی استانداردسازی‌شده که داده‌های ناهمگن گذشته (پیکربندی‌های JSON[۶۲]، پیاده‌سازی‌های کد، سنجه‌های عددی) را به نمایش‌های یکنواخت زبان طبیعی تبدیل می‌کنند تا پردازش و قیاس توسط مدل تسهیل شود [۳۹]. شکل ۳-۵ این رویکردها را به‌مثابه طیفی از راهبردهای یادگیری درون‌متنی نشان می‌دهد که استفاده از تاریخچه بهینه سازی را شامل می‌شود [۳۷].

[56] actionable feedback
[57] In-Context Learning from Optimization History
[58] optimization
[59] serialization
[60] chat-style dialogues
[61] few-shot demonstrations
[62] JavaScript Object Notation

در این رویکرد تکمیلی، ابرپارامترهای عددی برای بهبود استدلال گسسته‌سازی[۶۳] می‌شوند (مثلاً به سطوح «کم»، «متوسط»، «زیاد»)، و تجربه‌های استانداردسازی‌شده با نهفتارسازی[۶۴] و نمایه‌سازی در پایگاه برداری پشتیبانی می‌شوند تا بازیابی مبتنی بر شباهت، چند برترین های نمونه مرتبط را برای نمایش درون‌متنی برگزینند. فراتر از درج خام تجربه‌ها، استخراج دانش برون‌خط[۶۵] از طریق خلاصه‌سازی تکرارشونده مبتنی بر مدل و اعتبارسنجی پسین[۶۶] روی وظایف کنارگذاشته، اصول طراحی سطح‌بالا و بازخورد قابل اقدام را تقطیر می‌کند؛ این دانش استخراج‌شده می‌تواند به‌صورت راهنمای سامانه یا الگوهای اندک‌نمونه در متن تزریق شود و حتی تولید راه‌حل تک‌نمونه‌ای[۶۷] برای وظایف نو را میسر سازد [۳۹].

این روش به‌ویژه در سناریوهای کم‌بودجه مؤثر است؛ جایی که پیشین‌های یادگرفته‌شده مدل و تجربه‌های استانداردسازی‌شده می‌توانند بدون داده تجربی فراوان مسیر اکتشاف را هدایت کنند. در بسترهای بهینه‌سازی بیزی، مشاهدات تاریخی، آغاز گرم، نمونه‌برداری نامزد[۶۸] و مدلسازی جانشین را به‌طور کامل از راه سریال‌سازی زبان طبیعی ارزیابی‌های پیشین شرطی‌سازی می‌کنند؛ و در حالی‌که در حالت‌های بی‌نمونه[۶۹] یا کم‌نمونه[۷۰] عمل می‌کنند، کارایی رقابتی در قیاس با روش‌های سنتی نشان می‌دهند [۲۸]. تجربه‌های استانداردسازی‌شده با فراهم‌سازی نمونه‌های مشابه تأییدشده و قواعد طراحی تقطیرشده، می‌توانند این مراحل را دقیق‌تر شروع تازه[۷۱] کنند و میدان جست‌وجو را به‌صورت هدایت‌شده منقبض سازند [۳۹].

محدودیت اصلی در بهینه‌سازی‌های بلندافق[۷۲] رخ می‌نماید؛ جایی‌که قیود پنجره زمینه مستلزم نگهداشت گزینشی یا فشرده‌سازی با از دست رفتن اطلاعات تاریخچه است و چه‌بسا الگوهای حیاتی برای پالایش مرحله پایانی را حذف کند. استانداردسازی[۷۳] تا حدی این معضل را با خلاصه‌های ساخت‌یافته متراکم و بازیابی هدفمند تخفیف می‌دهد، اما همچنان

[63] discretized
[64] embed
[65] offline knowledge elicitation
[66] post-validation
[67] one-shot solution generation
[68] candidate sampling
[69] zero-shot
[70] few-shot
[71] warmstart
[72] long-horizon
[73] canonicalization

با برش اطلاعاتی، سوگیری‌های ناشی از گسسته‌سازی، و حساسیت به امتیازدهی ارتباط[۷۴] و پوشش مخزن مواجه است. با این‌همه، چون مصرف نهایی این دانش درون همان پنجره زمینه صورت می‌گیرد، مرز میان «دانش درون‌متنی صرف» و «تقویت مبتنی بر بازیابی» کم‌رنگ‌تر می‌شود؛ و ادغام تاریخچه آزمون با تجربه‌های استانداردسازی‌شده، پایایی و کارایی یادگیری درون‌متنی را در عمل ارتقا می‌دهد [۲۶، ۲۷، ۳۰، ۲۸، ۳۹].

۳-۴-۲ دانش بیرونی: بازیابی از ادبیات و مخازن[۷۵]

سامانه‌های پیشرفته‌تر، تولید تقویت‌شده با بازیابی را برای ادغام دانش بیرون از مسیر بهینه‌سازی به کار می‌گیرند. این چارچوب‌ها مخازن دانش ساخت‌یافته عموماً پایگاه‌های داده برداری نمایه‌شده با نهفتارها نگه می‌دارند تا بر پایه زمینه وظیفه جاری، اطلاعات مرتبط را بازیابی کرده و در راهنماهای متنی تزریق کنند و بدین‌سان تصمیم‌سازی را اطلاع‌رسانی کنند.

استخراج دانش مبتنی بر ادبیات پژوهشی[۷۶]

چند رویکرد، دانش راهبردی را از ادبیات علمی برای هدایت تصمیم‌های معماری گردآوری می‌کنند. یک راهبرد از کارگزاران خوانش تخصصی[۷۷] بهره می‌برد که مقالات اخیر را خزش[۷۸] کرده، نکته‌های روش‌شناختی را از چکیده‌ها و بخش‌های روش استخراج می‌کنند و در پایگاه‌های داده برداری برای بازیابی مبتنی بر شباهت بایگانی می‌نمایند [۳۶]. در خلال بهینه‌سازی، پیشنهادهای تغییر به‌منزله پرسش، اصول طراحی مرتبط را فراخوانی می‌کنند؛ و بدین‌ترتیب سامانه بدون آن‌که مدل پایه الزاماً بر تازه‌ترین انتشارات آموزش دیده باشد، از مرز دانش روز بهره می‌گیرد. نمونه‌ای دیگر، خلاصه‌هایی از مقالات arXiv و جست‌وجوهای وب را از طریق رابط‌های برنامه‌نویسی کاربردی[۷۹] بازیابی کرده و راهنماهای برنامه‌ریزی

[74] relevance scoring
[75] External Knowledge via Retrieval
[76] Literature-Driven Knowledge Extraction
[77] specialized reader agents
[78] crawl
[79] application programming interfaces (APIs)

را با بینش‌های بیرونی پیرامون مدل‌ها، ابرپارامترها و داده‌مجموعه‌ها غنی می‌کند تا تنوع و سازگاری طرح را ارتقا دهد [۳۸].

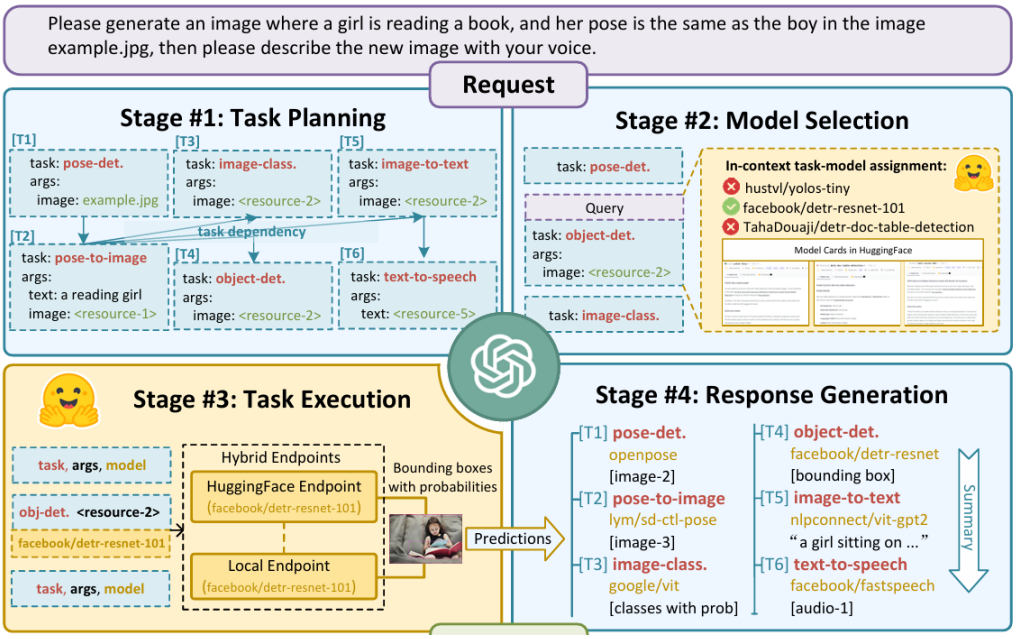
این استخراج دانش، برای طراحی معماری‌های عصبی بس سودمند است؛ چراکه نوآوری‌های اخیر در ترکیب لایه‌ها، اتصالات پرشی[۸۰] یا طرحواره‌های نرمال‌سازی[۸۱] چه‌بسا در پارامترهای منجمد[۸۲] مدل بازتاب نیافته باشند. با این‌همه، کیفیت دانش بازیابی‌شده به‌نحو حساس به سازوکار امتیازدهی ارتباط و پوشش پیکره ادبیات نمایه‌شده وابسته است.

مخازن داده‌مجموعه و مدل[۸۳]

در کنار بازیابی مبتنی بر ادبیات، چند سامانه از مخازن بیرونی برای فراداده‌های داده‌مجموعه‌ها و مدل‌های ازپیش‌آموزش‌دیده پرس‌وجو می‌کنند. چارچوب‌هایی که کل زنجیره خودکارسازی یادگیری ماشین را راهبری می‌کنند، کارت‌های داده‌مجموعه از پلتفرم‌هایی مانند Kaggle و کارت‌های مدل از HuggingFace را بازیابی کرده و فراداده ساخت‌یافته از جمله وجه‌های داده[۸۴]، متغیرهای هدف، معماری‌های مدل و بازه‌های ابرپارامتر را در راهنماهای متنی می‌گنجانند تا تصمیم‌های پایین‌دستی را غنی کنند (شکل ۳-۶) [۳۸، ۳۴]. برای داده‌مجموعه‌های نادیده، سازوکارهای انتقال مبتنی بر شباهت[۸۵] با محاسبه همبستگی میان کدگذاری کارت‌های داده (با مدل‌هایی مانند CLIP) مسائل مشابه را شناسایی کرده و ابرپارامترها یا الگوهای معماری را از تجربه‌های تاریخی منتقل می‌سازند [۳۳].

این تقویت مبتنی بر فراداده[۸۶] تعمیم‌پذیری میان حوزه‌های گوناگون را بدون نیاز به آموزش‌های خاص وظیفه ممکن می‌کند؛ هرچند به دسترس‌پذیری مخازن خوش‌سامان و برچسب‌گذاری دقیق فراداده متکی است.

[80] skip connections
[81] normalization schemes
[82] frozen
[83] Dataset and Model Repositories
[84] modalities
[85] similarity-based transfer
[86] metadata-driven augmentation



شکل ۳-۶: چارچوب HuggingGPT که از مخازن مدل HuggingFace برای خودکارسازی وظایف یادگیری ماشین بهره می‌گیرد [۳۴].

۳-۴-۳ راهبردهای تقویت آمیخته[87]

سامانه‌های پیشرفته روز، غالباً چند منبع دانش را برای بهره‌گیری از قوت‌های مکمل با هم ترکیب می‌کنند. چارچوب‌های چندکارگزاره ممکن است برنامه‌ریزی تقویت‌شده با بازیابی[88] که دانش ادبیات و مخازن را برای راهبردهای سطح‌بالا به کار می‌گیرد را با حافظه رویدادی برآمده از گزارش‌های تجربی که اجرای عملی را صیقل می‌دهد، جفت کنند [۳۶، ۳۸]. به‌همین سیاق، تاریخچه آزمون درون‌متنی می‌تواند با تجربه‌های استانداردسازی‌شده بازیابی‌شده غنی شود تا گرم‌آغاز بهینه‌سازی را به‌ویژه در مواجهه با وظایف نو با ارزیابی‌های اولیه محدود تسریع کند [۳۹].

گزینش معماری تقویت دانش، به‌طرز حساس با روش عملیاتی کارگزار برهم‌کنش دارد: دستوردهی تکرارشونده بیشترین سود را از خلاصه‌های فشرده تاریخی می‌برد؛ عملگرهای تکاملی برای نگهداشت تنوع به بایگانی‌های کیفیت–تنوع اتکا دارند؛ و کنترل‌گرهای جریان‌کار[89]

[87] Hybrid Augmentation Strategies
[88] retrieval-augmented planning
[89] workflow controllers

برای هماهنگ‌سازی مراحل ناهمگون خط لوله، به فراداده ساخت‌یافته نیازمندند. با گسترش ظرفیت‌های پنجره زمینه و پختگی سازوکارهای بازیابی، مرز میان دانش درون‌متنی و بیرونی هرچه بیشتر محو می‌شود و ادغامی غنی‌تر از پیشین‌های آموخته، شواهد تجربی و خبرگی حوزه را امکان‌پذیر می‌سازد.

۳-۵ تحلیل بر مبنای قالب خروجی مدل

۳-۵-۱ خروجی‌های سبک واژنامه‌ای

نمایش‌های ساخت‌یافته کلید—مقدار[۹۰] کدگذاری بی‌ابهام فراپارامترها یا انتخاب‌های معماری را با خوانایی ماشینی مستقیم فراهم می‌کنند. سامانه‌هایی که پیکربندی‌های قالب JSON تولید می‌کنند، پارامترهایی مانند نرخ یادگیری، اندازه دسته و ابعاد شبکه را مشخص می‌سازند [۲۶، ۳۷]. گونه‌های پیشرفته، استدلال زنجیره تفکر[۹۱] را پیش از خروجی ساخت‌یافته می‌گنجانند تا کیفیت پیشنهادها را با استدلال میانی صریح بهبود دهند، در حالی‌که مشخصات نهایی همچنان قابل تجزیه باقی می‌ماند. محدودیت اصلی، مقیدشدن اکتشاف به طرحواره‌های ازپیش‌تعریف‌شده[۹۲] است که می‌تواند کشف الگوهای طراحی نو را محدود کند. شکل ۳-۷ نمونه‌ای از خروجی قالب JSON را نشان می‌دهد.

۳-۵-۲ تولید کد برنامه

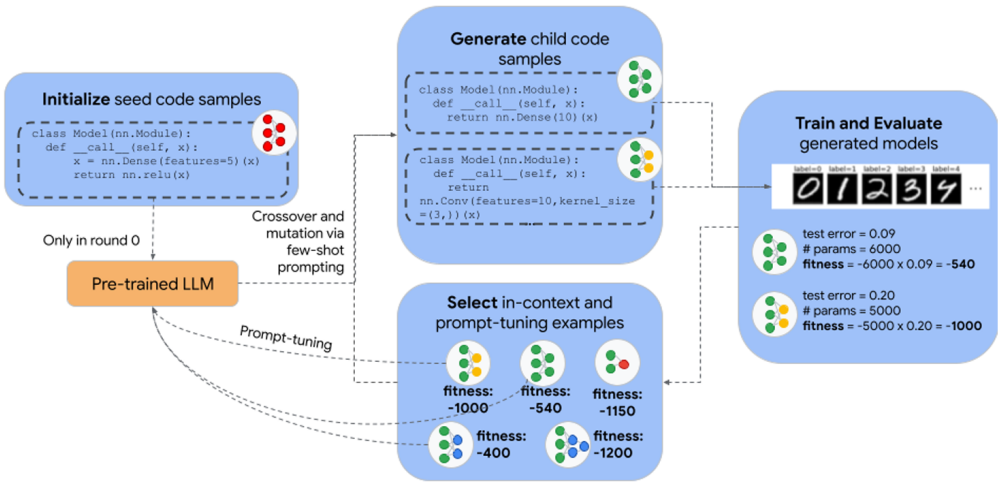
خروجی مبتنی بر کد با تکیه بر بیان‌پذیری کامل زبان‌های برنامه‌نویسی، از قیود فضاهای پیکربندی ازپیش‌تعریف‌شده می‌گریزد. چندین سامانه پیاده‌سازی‌های Python از شبکه‌های عصبی و خطوط لوله یادگیری ماشین را به‌صورت برنامه‌های کامل یا مولفه‌های ماژولار تولید می‌کنند و با آزمون‌های واحد خودکار سازگاری را می‌سنجند [۳۵، ۲۹، ۳۰]. برخی، تولید را بر معیارهای هدف با دستوردهی چندنمونه و نمونه‌هایی برگرفته از جمعیت‌های

[90] structured key-value representations
[91] chain-of-thought
[92] predefined schemas

```
Config: {"alpha": 0.1, "batch_size": 8, "depth": 1, "
    ↪ learning_rate_init": 0.0001, "width": 128}Analysis: The
    ↪ initial configuration with alpha=0.001, batch_size=32,
    ↪ depth=3, learning_rate_init=0.001, and width=64 achieved
    ↪ the lowest loss of 0.058537. The subsequent configuration
    ↪  with alpha=0.01, batch_size=16, depth=2,
    ↪ learning_rate_init=0.01, and width=256 resulted in a
    ↪ higher loss of 0.082927. It seems that decreasing the
    ↪ learning rate and increasing the width did not improve
    ↪ the performance. The latest configuration with alpha=0.1,
    ↪  batch_size=8, depth=1, learning_rate_init=0.0001, and
    ↪ width=128 also resulted in a higher loss of 0.13171.
```

شکل ۳-۷: نمونه‌ای از خروجی قالب JSON برای پیکربندی بهینه‌سازی فراپارامتر در سامانه مبتنی بر مدل زبانی بزرگ [۲۶]

ارزیابی‌شده شرطی می‌کنند؛ برخی دیگر، عملگرهای تکاملی مانند جهش و ترکیب را مستقیماً بر نمایش‌های کدی اعمال می‌کنند. گونه‌های ترکیبی، تولید کد برای پیاده‌سازی معماری را با فهرست‌های ساخت‌یافته فراپارامتر و متن قالب‌بندی‌شده گزارش وقایع[۹۳] آموزشی پیش‌بینی‌شده درمی‌آمیزند [۳۳، ۳۸]. شکل ۳-۸ نمونه‌ای از خروجی تولید کد را نشان می‌دهد. تولید کد، انعطاف طراحی را به حداکثر می‌رساند و امکان جست‌وجو در معماری‌های



شکل ۳-۸: نمونه‌ای از خروجی تولید کد در سامانه Evoprompting که از مدل زبانی بزرگ برای تولید و بهینه‌سازی کدهای Python شبکه‌های عصبی استفاده می‌کند [۳۰]

نامقید را بدون تعریف صریح اجزای ابتدایی فراهم می‌آورد. بااین‌حال، این رویکرد چالش‌های

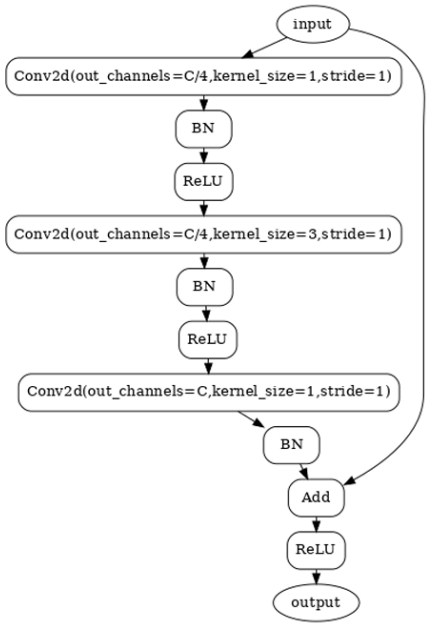
[93] log

اعتبارسنجی به همراه دارد: درستی نحوی[۹۴] لزوماً آموزش‌پذیری، رعایت قیود منابع[۹۵] یا معناداری معنایی[۹۶] را تضمین نمی‌کند. ازاین‌رو، سامانه‌ها به محیط‌های اجرا برای ارزیابی نیاز دارند و سازوکارهای مدیریت خطا را برای مواجهه با خروجی‌های نامعتبر پیاده می‌کنند؛ از جمله دستوردهی مجدد[۹۷] با استفاده از پیام‌های خطا به‌عنوان بازخورد.

۳-۵-۳ خروجی‌های درخت‌ساختار

نمایش‌های گراف/درخت بر روابط ترکیبی درون معماری‌ها تأکید می‌کنند و برای وظایفی که به تعیین صریح توپولوژی[۹۸] نیاز دارند سودمندند؛ بی‌آن‌که جزئیات پیاده‌سازی کدی که می‌تواند از استدلال ساختاری منحرف کند تحمیل شود. برخی سامانه‌ها نمایش متنی گراف جهت‌دار بدون‌دور[۹۹] با گره‌های شماره‌گذاری‌شده برای عملیات و اتصالات برمی‌گزینند [۳۶]؛ برخی دیگر از کدگذاری رشته‌ای جداکننده‌محور[۱۰۰] برای ویژگی‌های لایه سازگار با تولید خودبازگشتی[۱۰۱] استفاده می‌کنند [۳۲]. خروجی‌ها معمولاً با فرایندهای تجزیه تخصصی به کد اجرایی تبدیل می‌شوند و ابزارهای راستی‌آزمایی گراف محاسباتی[۱۰۲] علاوه بر صحت نحوی، همریختی[۱۰۳] با طرح‌های موجود را نیز می‌سنجند. شکل ۳-۹ نمونه‌ای از خروجی درخت‌ساختار را نشان می‌دهد. این قالب‌ها استدلال ترکیبی و راهبردهای تغییر سلسله‌مراتبی را تسهیل می‌کنند و به مدل امکان می‌دهند بر توپولوژی معماری مستقل از جزئیات پیاده‌سازی تمرکز کند. به‌کارگیری آن‌ها به طرح‌های کدگذاری حوزه‌ای[۱۰۴] و رویه‌های اعتبارسنجی ویژه نیاز دارد، اما با کاهش پیچیدگی وظیفه تولید از طریق سطح تجرید مناسب، کیفیت تولید را بهبود می‌بخشد.

[94] syntactic correctness
[95] resource constraints
[96] semantic meaningfulness
[97] re-prompting
[98] explicit topology specification
[99] Directed Acyclic Graph (DAG)
[100] delimited string encodings
[101] autoregressive generation
[102] computational graph
[103] isomorphism
[104] domain-specific encoding schemes



##ResNetBottleBlock##
0 : input
1 : output
2 : Conv2d(out_channels=C/4,kernel_size=1,stride=1)
3 : BN
4 : ReLU
5 : Conv2d(out_channels=C/4,kernel_size=3,stride=1)
6 : BN
7 : ReLU
8 : Conv2d(out_channels=C,kernel_size=1,stride=1)
9 : BN
10: Add
11: ReLU

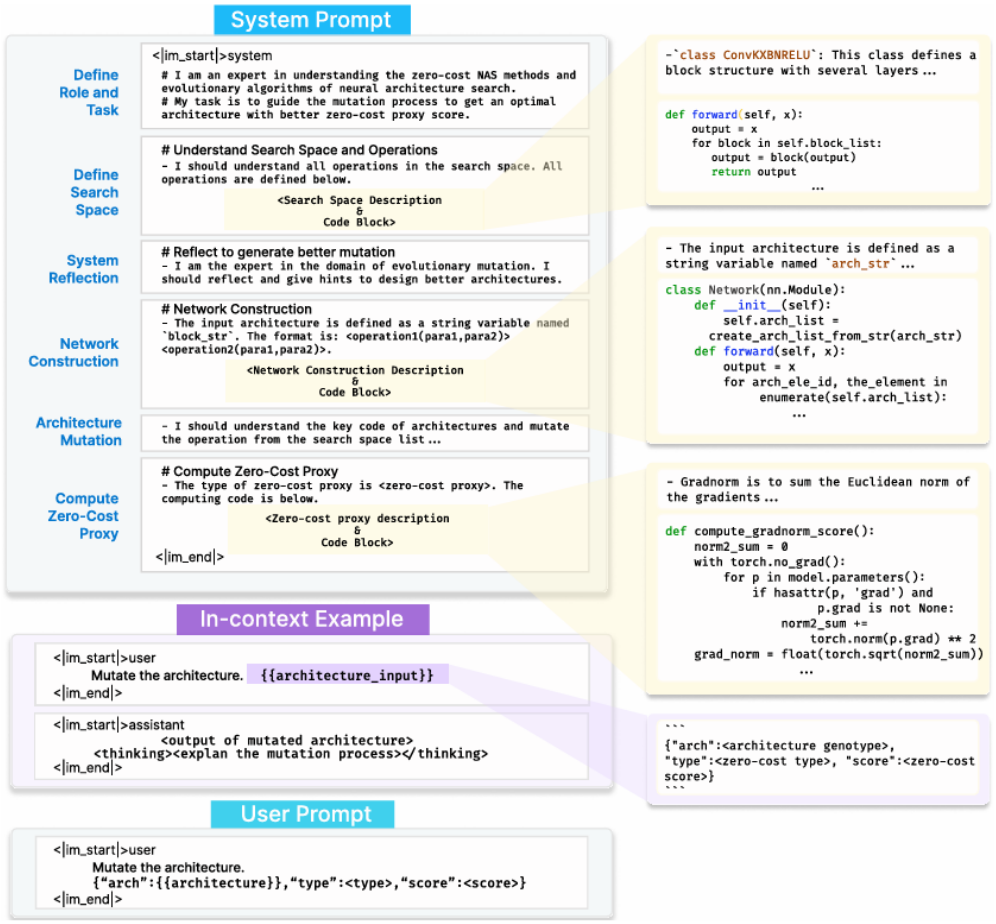
0 → 2
2 → 3
3 → 4
3 → 4
4 → 5
5 → 6
6 → 7
7 → 8
8 → 9
9 → 10
0 → 10
11 → 1

شکل ۳-۹: نمونه‌ای از نمایش گراف‌محور معماری شبکه عصبی. سمت چپ: تصویرسازی گراف جهت‌دار بدون‌دور . سمت راست: نمایش متنی گراف جهت‌دار بدون‌دور برای فهم مدل زبانی بزرگ. [۳۶]

۳-۵-۴ خروجی‌های ترکیبی

سامانه‌های پیشرفته، چندین گونه خروجی[۱۰۵] را ترکیب می‌کنند تا از قوت‌های مکمل قالب‌های مختلف در مراحل گوناگون خط لوله بهره ببرند. الگوی رایج، مشخصات ساخت‌یافته را با توضیحات زبان طبیعی همراه می‌کند تا هم اجرای ماشینی و هم تفسیر انسانی میسر شود [۳۷، ۲۶]. طرح‌های مفصل‌تر برای مراحل متمایز از قالب‌های متفاوت بهره می‌گیرند: JSON برای نیازمندی‌های قابل اعتبارسنجی صوری، زبان طبیعی برای برنامه‌ریزی و تحلیل منعطف، و کد اجرایی برای پیاده‌سازی‌های نهایی [۳۸، ۳۳]. برخی سامانه‌ها کدگذاری‌های ساختاری فشرده را در توضیحات زبان طبیعی می‌گنجانند تا مشخصات دقیق را با استدلال‌های قابل تفسیر تلفیق کنند [۳۱، ۳۶]. شکل ۳-۱۰ نمونه‌ای از خروجی ترکیبی را نشان می‌دهد. این روش بازتاب این واقعیت است که هیچ قالب یگانه‌ای به‌تنهایی برای همه جنبه‌های یادگیری

[105] output modalities



شکل ۳-۱۰: نمونه‌ای از خروجی ترکیبی در سامانه RZNAS که از قالب‌های JSON و کد Python برای طراحی معماری شبکه عصبی استفاده می‌کند [۳۱]

ماشین خودکار بهینه نیست: خروجی ساخت‌یافته برای تجزیه و اعتبارسنجی مناسب است؛ کد، پیاده‌سازی منعطف را ممکن می‌سازد؛ گراف‌ها استدلال ترکیبی را تقویت می‌کنند؛ و زبان طبیعی تفسیرپذیری و زمینه غنی را فراهم می‌کند و با موارد دشوار صوری‌سازی روبه‌رو می‌شود. ادغام موفق مستلزم طراحی دقیق گذارهای بین قالب‌ها[۱۰۶]، رویه‌های اعتبارسنجی میان‌گونه‌ای[۱۰۷] و راهبردهایی برای مدیریت ناهمخوانی[۱۰۸] در صورت تعارض نمایش‌ها است.

[106] format transitions
[107] across modalities
[108] inconsistencies

عنوان	عامل	روش سنتی	خروجی	دانش خارجی	فضای جستجو	بدون آموزش	وظیفه
LLM for HPO [۲۶]	تک	—	واژنامه	×	اختیاری	×	HPO
GENIUS [۲۷]	تک	—	واژنامه	×	بله	×	NAS
LLMATIC [۲۹]	تک	EA	کد	×	خیر	×	NAS
LLAMA-NAS [۴۰]	تک	EA	واژنامه	×	بله	×	NAS
Text to ML [۳۵]	چند	—	کد	×	خیر	✓	AutoML
AgentHPO [۳۷]	چند	—	واژنامه	×	بله	×	HPO
AutoML-Agent [۳۸]	چند	—	کد	✓	خیر	✓	AutoML
LLAMBO [۲۸]	چند	BO	متن	×	بله	✓	HPO
Nader [۳۶]	چند	—	درخت	✓	بله	×	NAS
RZ-NAS [۳۱]	تک	EA	کد†	×	بله	✓	NAS
EvoPrompting [۳۰]	تک	EA	کد	×	خیر	×	NAS
AutoML-GPT [۳۳]	تک	—	کد	×	بله	×	AutoML
HuggingGPT [۳۴]	چند	—	کد	✓	خیر	×	AutoML
GPT-NAS [۳۲]	تک	EA	درخت	×	خیر	×	NAS
ML Copilot [۳۹]	چند	—	متن	✓	خیر	×	AutoML

جدول ۳-۱: مقایسه فشرده مقالات مبتنی بر LLM. EA=الگوریتم‌های تکاملی، BO=بهینه‌سازی بیزی. نشانه‌ها: †=تولید کد + واژنامه.

۳-۶ طبقه بندی کار های مرتبط

همانطور که در جدول ۳-۱ مشاهده می‌شود، روش‌های بررسی شده در محور معماری عامل تقریباً به طور مساوی بین ساختارهای تک‌عاملی (۸ مورد) و چندعاملی (۷ مورد) تقسیم شده‌اند. برخلاف تصور اولیه، اکثر قریب به اتفاق روش‌ها (۱۱ مورد از ۱۵) در حال حاضر بر دانش درونی (مانند تاریخچه بهینه‌سازی) تکیه می‌کنند و تنها اقلیت کوچکی (۴ مورد) به صراحت از دانش خارجی (مانند مقالات علمی از طریق تولید تقویت‌شده با بازیابی) بهره می‌برند.

بیشترین تنوع در قالب خروجی دیده می‌شود؛ جایی که پژوهشگران از قالب‌های گوناگونی متناسب با نیاز مسئله استفاده کرده‌اند. این خروجی‌ها از کدهای برنامه (مناسب برای پیاده‌سازی مستقیم و انعطاف‌پذیری بالا)، واژه‌نامه‌های ساختاریافته (مفید برای پیکربندی‌های صریح)، نمایش‌های درختی (برای تعریف توپولوژی) و حتی متن زبان طبیعی متغیر هستند.

فصل ۴

نتیجه گیری و کارهای آینده

۴-۱ نتیجه‌گیری

این گزارش سمینار به بررسی و تحلیل رویکردهای نوظهور در یادگیری ماشین خودکار پرداخت که از قابلیت‌های مدل‌های زبانی بزرگ در قالب سیستم‌های عامل‌ـ‌محور بهره می‌برند. مرور ادبیات نشان داد که این حوزه به سرعت در حال فاصله گرفتن از بهینه‌سازهای جعبه‌سیاه سنتی و حرکت به سوی روندهای بر اساس دانش، تفسیرپذیر و خودمختار است.

ما روش‌های موجود را از سه منظر کلیدی طبقه‌بندی کردیم: معماری عامل (تک‌عاملی در برابر چندعاملی)، منابع دانش (درونی در برابر بیرونی) و قالب خروجی (ساختاریافته، کد، یا ترکیبی).

یافته‌ها حاکی از آن است که عامل‌های تک‌عاملی، به‌ویژه آن‌هایی که از دستوردهی تکراری یا عملگرهای تکاملی استفاده می‌کنند، برای بهینه‌سازی ابرپارامترها و جستجوی معماری عصبی محدود مؤثر هستند. در مقابل، معماری‌های چندعاملی با تفکیک نقش (مانند پژوهشگر و توسعه‌دهنده)، پتانسیل بیشتری برای مدیریت خطوط لوله پیچیده و بلندـ‌افق یادگیری ماشین خودکار از خود نشان می‌دهند.

ادغام تولید تقویت‌شده با بازیابی یک پیشرفت کلیدی است که به عامل‌ها اجازه می‌دهد

تا از دانش ایستای خود فراتر رفته و از ادبیات پژوهشی، مخازن کد[1] و نتایج آزمایش‌های گذشته برای اتخاذ تصمیمات آگاهانه‌تر استفاده کنند. در نهایت، قالب خروجی نشان‌دهنده یک توازن میان خوانایی ماشینی[2] (مانند JSON) و بیانگری (مانند تولید کد کامل) است، که رویکردهای ترکیبی به عنوان راه‌حلی میانه در حال ظهور هستند. در مجموع، یادگیری ماشین خودکار عامل‌ـ‌محور یک حوزه تحقیقاتی بسیار پویا است که نویدبخش خودکارسازی هوشمندانه‌تر و کارآمدتر فرایندهای علم داده است.

۴-۲ مسائل باز و کارهای قابل انجام

حوزه یادگیری ماشین خودکار مبتنی بر عامل‌های زبانی، با وجود پیشرفت‌های سریع، همچنان در مراحل اولیه تکامل خود قرار دارد و مملو از مسائل باز و زمینه‌های پژوهشی است. بر اساس تحلیل‌های ارائه‌شده در این گزارش، می‌توان کارهای آتی را در چند محور اصلی دسته‌بندی کرد:

بهینه‌سازی برای منابع محدود و افزایش کارایی: اکثر روش‌های فعلی بر مدل‌های زبانی بزرگ و پرهزینه (مانند خانواده های GPT, Claude, Gemini) متکی هستند. یک مسئله باز کلیدی، تطبیق این رویکردها برای سناریوهایی با منابع محاسباتی محدود است. کارهای آینده می‌تواند شامل تحقیق بر روی ریزتنظیم کردن مدل‌های زبانی با پارامتر کم (مثلاً Gemma-3-12B یا مدل‌های کوچک‌تر) برای وظایف خاص یادگیری ماشین خودکار مانند بهینه‌سازی ابرپارامترها باشد تا به جای اتکای صرف به یادگیری درون‌ـ‌متنی، دانش تخصصی مستقیماً در پارامترهای مدل تزریق شود؛ همچنین استفاده از تکنیک‌های تقطیر دانش[3] برای انتقال قابلیت‌های استدلال یک مدل زبانی بزرگ به یک مدل کوچک‌تر و کارآمدتر که بتواند به عنوان یک عامل یادگیری ماشین خودکار سبک عمل کند، مسیر مهمی به شمار می‌رود.

یکپارچه‌سازی و مدیریت دانش پیشرفته: همانطور که در تحلیل‌ها مشاهده شد، بسیاری از سیستم‌ها هنوز به دانش درونی (تاریخچه بهینه‌سازی) محدود هستند. غنی‌سازی عامل‌ها

[1] code repositories
[2] machine-readability
[3] knowledge distillation

با دانش خارجی یک مرز پژوهشی مهم است، از جمله طراحی پایگاه‌های دانش پویا و خود-بهبودگر که نه تنها مقالات علمی را شامل شوند، بلکه از مخازن کد (مانند GitHub)، بحث‌های فنی (مانند Stack Overflow) و نتایج بنچمارک‌های عمومی (مانند Papers with Code) نیز تغذیه شوند؛ نیز توسعه راهبردهایی برای مدیریت دانش متناقض، به این صورت که اگر یک مقاله روشی را پیشنهاد دهد اما نتایج تجربی عامل چیز دیگری را نشان دهد، عامل بتواند این تضاد را حل کند.

سیستم‌های چندعاملی با تیم‌های پژوهشی موازی (مسابقه عامل‌ها برای ریزتنظیم): طراحی یک چارچوب که در آن چند «تیم» عامل (با نقش‌های متمایز مانند پژوهشگر، مهندس داده و مربی) هر کدام یک مدل را برای یک تسک مشخص فاین‌تیون کنند و در نهایت بهترین مدل انتخاب یا تجمیع شود، مستلزم وجود هماهنگ‌کننده مرکزی برای تشکیل تیم‌ها، تخصیص نقش‌ها، تعریف قرارداد رابط (مشخصات داده، بودجه، محدودیت‌ها) و زمان‌بندی اجرای آزمایش‌ها است؛ علاوه بر این، باید راهبرد انتخاب یا تجمیع مشخص شود تا بر اساس معیارهای چندهدفه (دقت، زمان، حافظه) بهترین مدل انتخاب گردد و در صورت نزدیک‌بودن عملکردها، Ensembling سبک (مثلاً logit averaging) و تحلیل جبهه پارتو ارزیابی شود؛ همچنین اشتراک دانش بین تیم‌ها از طریق حافظه مشترک مبتنی بر بازیابی (RAG) برای دسترسی به یافته‌ها، تنظیمات موفق و خطاهای رایج و استانداردسازی ثبت آزمایش‌ها جهت تکرارپذیری اهمیت دارد؛ و نهایتاً تضمین ایمنی و بازتولیدپذیری با تثبیت بذر تصادفی، قفل‌کردن نسخه وابستگی‌ها، نظارت بر نشتی داده و تعریف آزمون‌های سلامت برای جلوگیری از خطاهای کدنویسی عامل‌ها ضروری است.

سامانه عامل‌محور برای انتخاب و تنظیم بهینه مدل در شرایط . تمرکز بر سه تصمیم کلیدی در انتقال یادگیری است: انتخاب مدل پایه، استراتژی ریزتنظیم و داده‌افزایی؛ و در این میان، تولید یک طرح پیکربندی ساختاریافته (JSON) به‌عنوان خروجی میانی پیشنهاد می‌شود. معماری چندعاملی شامل عامل تحلیل‌گرِ دیتاست، عامل استراتژیست برای برنامه‌ریزی و عامل اجراکننده است که به موتور اجرا متصل می‌شود تا پیکربندی تولیدشده را به کد قابل‌اعتماد تبدیل کند.

۴-۳ معرفی موضوع مورد نظر برای پایان نامه

با توجه به تحلیل‌های صورت‌گرفته در این سمینار و بررسی مسائل باز موجود، موضوع زیر که بر اساس ایده سوم پیشنهادی تدوین شده است، به عنوان یک حوزه پژوهشی نوآورانه، کاربردی و دارای عمق کافی برای یک پایان‌نامه کارشناسی ارشد انتخاب می‌گردد.

عنوان پیشنهادی

طراحی و پیاده‌سازی یک سیستم یادگیری ماشین خودکار عامل‌ـ محور برای انتخاب و تنظیم بهینه مدل

در بسیاری از کاربردهای عملی یادگیری عمیق، دسترسی به داده‌های برچسب‌دار انبوه امکان‌پذیر نیست. در چنین شرایطی، رویکرد غالب، استفاده از انتقال یادگیری[۴] از طریق ریزتنظیم کردن مدل‌های از پیش‌آموزش‌دیده است. با این حال، موفقیت این رویکرد به شدت به مجموعه‌ای از تصمیمات پیچیده و به هم وابسته بستگی دارد:

انتخاب مدل پایه[۵]: کدام مدل از میان ده‌ها مدل موجود در استخر مدل ها (مثلاً ResNet، ViT، EfficientNet) برای دیتاست و تسک مورد نظر مناسب‌تر است؟

انتخاب استراتژی ریزتنظیم: آیا باید کل مدل را ریزتنظیم کرد، فقط لایه‌های آخر را آموزش داد، یا از روش‌های کارآمد پارامتری مانند LoRA و دیگر تکنیک‌های PEFT استفاده نمود؟

انتخاب روش‌های پیش‌پردازش و داده‌افزایی: کدام تکنیک‌های داده‌افزایی (مانند CutMix، Mixup، RandAugment) با توجه به ویژگی‌های دیتاست، بیشترین بهبود را در عملکرد مدل ایجاد می‌کنند؟

فضای جستجوی حاصل از ترکیب این انتخاب‌ها بسیار بزرگ و گسسته است و روش‌های سنتی یادگیری ماشین خودکار برای کاوش مؤثر در آن با چالش مواجه هستند. این پژوهش قصد دارد با بهره‌گیری از یک سیستم چندعاملی مبتنی بر مدل زبانی بزرگ، این فرآیند

[4] Transfer Learning
[5] backbone

تصمیم‌گیری را خودکار و هوشمند سازد. هدف اصلی، ساخت عاملی است که بتواند با تحلیل ویژگی‌های دیتاست ورودی، یک طرح اجرایی[۶] بهینه تولید کند که بهترین ترکیب از مدل، روش ریزتنظیم و تکنیک‌های داده‌افزایی را برای دستیابی به حداکثر دقت با حداقل منابع ممکن، مشخص نماید.

طراحی معماری عامل‌ـ‌محور: یک گردش کار مبتنی بر عامل‌های مدل زبانی بزرگ طراحی می‌شود که وظایف را به صورت منطقی تقسیم کند؛ این معماری می‌تواند شامل یک "عامل تحلیل‌گر" برای استخراج فراداده از دیتاست، یک "عامل استراتژیست" برای تولید طرح پیکربندی بر اساس تحلیل‌ها و دانش پیشین، و یک "عامل اجراکننده" برای اجرای کد مبتنی بر کانفیگ تولید شده باشد.

توسعه مکانیزم تصمیم‌گیری مبتنی بر مدل زبانی بزرگ: مهندسی دستور[۷] و طراحی ساختار ورودی/خروجی به گونه‌ای صورت می‌گیرد که مدل زبانی بزرگ بتواند بر اساس ویژگی‌های دیتاست (مانند اندازه، تعداد کلاس‌ها، نوع داده) و محدودیت‌های منابع، استدلال کرده و تصمیمات آگاهانه بگیرد. خروجی مدل زبانی بزرگ یک فایل پیکربندی ساختاریافته (مثلاً در قالب JSON) خواهد بود.

پیاده‌سازی پل ارتباطی بین استدلال و اجرا: یک موتور اجرایی ساخته می‌شود که فایل پیکربندی تولیدشده توسط مدل زبانی بزرگ را تفسیر کرده و آن را به کد پایتون قابل اجرا تبدیل و اجرا نماید. این رویکرد، استدلال سطح بالای مدل زبانی بزرگ را از اجرای سطح پایین و مستعد خطای کد جدا می‌کند.

ارزیابی جامع سیستم: عملکرد سیستم پیشنهادی بر روی چندین دیتاست بنچمارک با اندازه‌های متفاوت ارزیابی می‌شود و نتایج (دقت و هزینه محاسباتی) با روش‌های پایه مانند انتخاب تصادفی، یک استراتژی ریزتنظیم ثابت و در صورت امکان، سایر ابزارهای یادگیری ماشین خودکار مقایسه خواهد شد.

این پژوهش در چند جنبه دارای نوآوری است: (۱) به جای تمرکز بر یک جزء منفرد مانند بهینه‌سازی ابرپارامترها یا جستجوی معماری عصبی، یک خط لوله کامل و یکپارچه

[6] Configuration Plan
[7] Prompt Engineering

برای انتقال یادگیری را هدف قرار می‌دهد. (۲) از مدل زبانی بزرگ نه به عنوان یک بهینه‌ساز جعبه‌ـ‌سیاه، بلکه به عنوان یک موتور استدلال و برنامه‌ریزی برای تولید یک طرح اجرایی شفاف و قابل تفسیر استفاده می‌کند. (۳) استفاده از یک فایل پیکربندی به عنوان خروجی میانی، یک راهکار نوین برای ترکیب قدرت استدلال مدل زبانی بزرگ با قابلیت اطمینان و استحکام سیستم‌های نرم‌افزاری کدمحور است. موفقیت این پروژه می‌تواند گام مهمی در جهت "مردمی‌سازی" استفاده بهینه از مدل‌های پایه باشد و به محققان و مهندسان کمک کند تا با سرعت و کارایی بیشتری به نتایج مطلوب دست یابند.

کتاب‌نامه

[1] I. Salehin, M. S. Islam, P. Saha, S. Noman, A. Tuni, M. M. Hasan, and M. A. Baten, "Automl: A systematic review on automated machine learning with neural architecture search," *Journal of Information and Intelligence*, vol.2, no.1, pp.52–81, 2024.

[2] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol.30, 2017.

[3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pp.4171–4186, 2019.

[4] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol.33, pp.1877–1901, 2020.

[5] L. Wang, C. Lyu, T. Ji, Z. Zhang, D. Yu, S. Shi, and Z. Tu, "Document-level machine translation with large language models," in *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing* (H. Bouamor, J. Pino, and K. Bali, eds.), (Singapore), pp.16646–16661, Association for Computational Linguistics, Dec. 2023.

[6] L. Gao, A. Madaan, S. Zhou, U. Alon, P. Liu, Y. Yang, J. Callan, and G. Neubig, "Pal: Program-aided language models," in *International Conference on Machine Learning*, pp.10764–10799, PMLR, 2023.

[7] L. Pan, A. Albalak, X. Wang, and W. Wang, "Logic-LM: Empowering large language models with symbolic solvers for faithful logical reasoning," in *Findings of the Association for Com-*

putational Linguistics: EMNLP 2023 (H. Bouamor, J. Pino, and K. Bali, eds.), (Singapore), pp.3806–3824, Association for Computational Linguistics, Dec. 2023.

[8] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao, "React: Synergizing reasoning and acting in language models," in *International Conference on Learning Representations (ICLR)*, 2023.

[9] L. Wang, C. Ma, X. Feng, Z. Zhang, H. Yang, J. Zhang, Z. Chen, J. Tang, X. Chen, Y. Lin, *et al.*, "A survey on large language model based autonomous agents," *Frontiers of Computer Science*, vol.18, no.6, p.186345, 2024.

[10] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, S. Riedel, and D. Kiela, "Retrieval-augmented generation for knowledge-intensive nlp tasks," in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, (Red Hook, NY, USA), Curran Associates Inc., 2020.

[11] Y. Xia, J. Zhou, Z. Shi, J. Chen, and H. Huang, "Improving retrieval augmented language model with self-reasoning," in *Proceedings of the AAAI conference on artificial intelligence*, vol.39, pp.25534–25542, 2025.

[12] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, H. Wang, and H. Wang, "Retrieval-augmented generation for large language models: A survey," *arXiv preprint arXiv:2312.10997*, vol.2, no.1, 2023.

[13] A. Singh, A. Ehtesham, S. Kumar, and T. T. Khoei, "Agentic retrieval-augmented generation: A survey on agentic rag," *arXiv preprint arXiv:2501.09136*, 2025.

[14] J. R. Rice, "The algorithm selection problem**this work was partially supported by the national science foundation through grant gp-32940x. this chapter was presented as the george e. forsythe memorial lecture at the computer science conference, february 19, 1975, washington, d. c.," vol.15 of *Advances in Computers*, pp.65–118, Elsevier, 1976.

[15] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of Machine Learning Research*, vol.13, no.10, pp.281–305, 2012.

[16] C. Thornton, F. Hutter, H. H. Hoos, and K. Leyton-Brown, "Auto-weka: combined selection and hyperparameter optimization of classification algorithms," KDD '13, (New York, NY, USA), p.847–855, Association for Computing Machinery, 2013.

[17] F. Hutter, H. H. Hoos, and K. Leyton-Brown, “Sequential model-based optimization for general algorithm configuration,” in *Learning and Intelligent Optimization* (C. A. C. Coello, ed.), (Berlin, Heidelberg), pp.507–523, Springer Berlin Heidelberg, 2011.

[18] M. Feurer, A. Klein, K. Eggensperger, J. Springenberg, M. Blum, and F. Hutter, “Efficient and robust automated machine learning,” in *Advances in Neural Information Processing Systems* (C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, eds.), vol.28, Curran Associates, Inc., 2015.

[19] R. S. Olson and J. H. Moore, “Tpot: A tree-based pipeline optimization tool for automating machine learning,” in *Proceedings of the Workshop on Automatic Machine Learning* (F. Hutter, L. Kotthoff, and J. Vanschoren, eds.), vol.64 of *Proceedings of Machine Learning Research*, (New York, New York, USA), pp.66–74, PMLR, 24 Jun 2016.

[20] K. Jamieson and A. Talwalkar, “Non-stochastic best arm identification and hyperparameter optimization,” in *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics* (A. Gretton and C. C. Robert, eds.), vol.51 of *Proceedings of Machine Learning Research*, (Cadiz, Spain), pp.240–248, PMLR, 09–11 May 2016.

[21] L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar, “Hyperband: A novel bandit-based approach to hyperparameter optimization,” *Journal of Machine Learning Research*, vol.18, no.185, pp.1–52, 2018.

[22] B. Bischl, G. Casalicchio, M. Feurer, P. Gijsbers, F. Hutter, M. Lang, R. G. Mantovani, J. N. van Rijn, and J. Vanschoren, “Openml benchmarking suites,” 2021.

[23] B. Zoph and Q. Le, “Neural architecture search with reinforcement learning,” in *International Conference on Learning Representations*, 2017.

[24] H. Pham, M. Guan, B. Zoph, Q. Le, and J. Dean, “Efficient neural architecture search via parameters sharing,” in *Proceedings of the 35th International Conference on Machine Learning* (J. Dy and A. Krause, eds.), vol.80 of *Proceedings of Machine Learning Research*, pp.4095–4104, PMLR, 10–15 Jul 2018.

[25] C. Ying, A. Klein, E. Christiansen, E. Real, K. Murphy, and F. Hutter, “NAS-bench-101: Towards reproducible neural architecture search,” in *Proceedings of the 36th International Conference on Machine Learning* (K. Chaudhuri and R. Salakhutdinov, eds.), vol.97 of *Proceedings of Machine Learning Research*, pp.7105–7114, PMLR, 09–15 Jun 2019.

[26] M. Zhang, N. Desai, J. Bae, J. Lorraine, and J. Ba, “Using large language models for hyperparameter optimization,” in *NeurIPS 2023 Foundation Models for Decision Making Workshop*, 2023.

[27] M. Zheng, X. Su, S. You, F. Wang, C. Qian, C. Xu, and S. Albanie, “Can gpt-4 perform neural architecture search?,” 2023.

[28] T. Liu, N. Astorga, N. Seedat, and M. van der Schaar, “Large language models to enhance bayesian optimization,” in *The Twelfth International Conference on Learning Representations*, 2024.

[29] M. U. Nasir, S. Earle, J. Togelius, S. James, and C. Cleghorn, “Llmatic: Neural architecture search via large language models and quality diversity optimization,” in *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO ’24, (New York, NY, USA), p.1110–1118, Association for Computing Machinery, 2024.

[30] A. Chen, D. Dohan, and D. So, “Evoprompting: Language models for code-level neural architecture search,” in *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[31] Z. Ji, G. Zhu, C. Yuan, and Y. Huang, “RZ-NAS: Enhancing LLM-guided neural architecture search via reflective zero-cost strategy,” in *Forty-second International Conference on Machine Learning*, 2025.

[32] C. Yu, X. Liu, Y. Wang, Y. Liu, W. Feng, X. Deng, C. Tang, and J. Lv, “Gpt-nas: Neural architecture search meets generative pre-trained transformer model,” *Big Data Mining and Analytics*, vol.8, no.1, pp.45–64, 2025.

[33] S. Zhang, C. Gong, L. Wu, X. Liu, and M. Zhou, “Automl-gpt: Automatic machine learning with gpt,” 2023.

[34] Y. Shen, K. Song, X. Tan, D. Li, W. Lu, and Y. Zhuang, “Hugginggpt: Solving ai tasks with chatgpt and its friends in huggingface,” in *Advances in Neural Information Processing Systems*, 2023.

[35] J. Xu, J. Li, Z. Liu, N. Suryanarayanan, G. Zhou, J. GUO, H. Iba, and K. Tei, “Large language models synergize with automated machine learning,” *Transactions on Machine Learning Research*, 2024.

[36] Z. Yang, W. Zeng, S. Jin, C. Qian, P. Luo, and W. Liu, "Nader: Neural architecture design via multi-agent collaboration," in *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, pp.4452–4461, June 2025.

[37] S. Liu, C. Gao, and Y. Li, "AgentHPO: Large language model agent for hyper-parameter optimization," in *The Second Conference on Parsimony and Learning (Proceedings Track)*, 2025.

[38] P. Trirat, W. Jeong, and S. J. Hwang, "AutoML-agent: A multi-agent LLM framework for full-pipeline autoML," in *Forty-second International Conference on Machine Learning*, 2025.

[39] L. Zhang, Y. Zhang, K. Ren, D. Li, and Y. Yang, "MLCopilot: Unleashing the power of large language models in solving machine learning tasks," in *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)* (Y. Graham and M. Purver, eds.), (St. Julian's, Malta), pp.2931–2959, Association for Computational Linguistics, Mar. 2024.

[40] A. Sarah, S. Nittur Sridhar, M. Szankin, and S. Sundaresan, "Llama-nas: Efficient neural architecture search for large language models," in *European Conference on Computer Vision*, pp.67–74, Springer, 2024.

واژه‌نامه فارسی به انگلیسی

آزمون‌ها ..... trials
آزمون‌های واحد خودکار ..... automated unit tests
آگاه از زمینه ..... context-aware
ابرپارامترها ..... Hyperparameters
ابزار ..... tool
اتصالات پرشی ..... skip connections
ادبیات پژوهشی ..... literature
استانداردسازی ..... canonicalization
استخراج دانش برون‌خط ..... offline knowledge elicitation
استخراج دانش مبتنی بر ادبیات پژوهشی ..... Literature-Driven Knowledge Extraction
استدلال تقویت‌شده با بازیابی ..... retrieval-augmented reasoning
اسناد ..... documents
اشتراک پارامتر ..... parameter sharing
اعتبارسنجی پسین ..... post-validation
الگوریتم ژنتیک ..... genetic algorithm
الگوریتم‌های تکاملی ..... evolutionary algorithms
الگوریتم‌های جستجو ..... search algorithms
الگوهای اندک‌نمونه ..... few-shot demonstrations
امتیازدهی ارتباط ..... relevance scoring
انتخاب مدل ..... model selection
انتقال مبتنی بر شباهت ..... similarity-based transfer
انتقال یادگیری ..... Transfer Learning
بازخورد زبانی ..... linguistic feedback
بازخورد قابل اقدام ..... actionable feedback
بازساز ..... reconstructor
بازیابی مبتنی بر شباهت ..... similarity-based retrieval

بازیابی مبتنی بر شباهت معنایی ..... semantic similarity retrieval
برنامه‌ریزی ..... planning
برنامه‌ریزی تقویت‌شده با بازیابی ..... retrieval-augmented planning
برنامه‌ریزی/تولید راه‌حل ..... planning/solution generation
برنامه‌نویسی ژنتیکی ..... genetic programming
بلندافق ..... long-horizon
بهینه‌سازی ..... optimization
بهینه‌سازی ابرپارامترها ..... Hyperparameter Optimization (HPO)
بهینه‌سازی بیزی ..... Bayesian optimization
به‌روز بودن ..... up-to-dateness
بی‌نمونه ..... zero-shot
تحلیل منابع دانش ..... Knowledge Source Analysis
ترجمه ماشینی ..... machine translation
ترکیب ..... crossover
تطبیق‌پذیر ..... adaptive
تعامل ..... interact
تعمیم‌پذیری ..... generalization
تعیین صریح توپولوژی ..... explicit topology specification
تفکر ..... Reasoning
تفکیک وظایف ..... task decomposition
تقطیر دانش ..... knowledge distillation
تقویت مبتنی بر فراداده ..... metadata-driven augmentation
تنظیم ابرپارامترها ..... hyperparameter tuning
تنظیم دستور ..... prompt-tuning
توانایی‌های بازتابی ..... reflective capabilities
توجه خودی ..... self-attention
تولید تقویت‌شده با بازیابی ..... Retrieval-Augmented Generation (RAG)
تولید خودبازگشتی ..... autoregressive generation
تولید راه‌حل تک‌نمونه‌ای ..... one-shot solution generation
تولید کد ..... code generation
توهم ..... hallucination
تک‌عاملی ..... Single-Agent
تیم توسعه ..... Development Team
تیم پژوهش ..... Research Team

جانشین surrogate
جستجوی تصادفی Random Search
جستجوی شبکه‌ای Grid Search
جستجوی معماری عصبی Neural Architecture Search (NAS)
جعبه‌سیاه black-box
جنگل‌های تصادفی random forests
جهش mutation
حافظه ترتیبی episodic memory
حسگرها sensors
حلقه‌های بازخورد feedback loops
حلقه‌های بازنگری revision loops
خزش crawl
خط لوله pipeline
خلاصه‌سازی متن text summarization
خوانایی ماشینی machine-readability
خودبازتابی self-reflection
خودمختار autonomous
دانش بیرونی: بازیابی از ادبیات و مخازن External Knowledge via Retrieval
دانش درونی: تاریخچه آزمون و بازتاب System-Internal Knowledge: Trials and Reflection
درستی نحوی syntactic correctness
درهم‌تنیده interleaved
درک زبان طبیعی Natural Language Understanding (NLU)
درک زمینه understanding context
دستور prompt
دستوردهی تکراری iterative prompting
دستوردهی زمینه‌مند contextual prompting
دستوردهی مجدد re-prompting
رابط‌های برنامه‌نویسی کاربردی application programming interfaces (APIs)
راهبردهای باندیتی bandit-based strategies
راهبردهای تقویت آمیخته Hybrid Augmentation Strategies
راهبردهای تقویت دانش knowledge augmentation strategies
ردپاهای استدلالی reasoning traces
رقابتی competitive
رمزگذار encoder

رمزگشا . . . . . decoder
رهگذر تفکیک کارکردی . . . . . functional decomposition
روش‌های تکاملی . . . . . evolutionary methods
رگرسیون . . . . . regression
ریزتنظیم . . . . . fine-tuning
زمینه . . . . . context
زنجیره تفکر . . . . . chain-of-thought
سازوکار بازیابی اطلاعات . . . . . information retrieval mechanism
سبک‌گپ . . . . . chat-style dialogues
سریال‌سازی . . . . . serialization
سیستم‌های عامل‌ـ محور . . . . . Agent-based Systems
شبکه‌های عصبی عمیق . . . . . Deep Neural Networks
شروع تازه . . . . . warmstart
شهود . . . . . intuition
طبقه‌بندی . . . . . classification
طرح اجرایی . . . . . Configuration Plan
طرحواره‌های ازپیش‌تعریف‌شده . . . . . predefined schemas
طرحواره‌های نرمال‌سازی . . . . . normalization schemes
طرح‌های کدگذاری حوزه‌ای . . . . . domain-specific encoding schemes
عامل . . . . . agent
عمل . . . . . Action
عملگرهای تکاملی . . . . . Evolutionary Operators
فرا‌ـ یادگیری . . . . . meta-learning
فراداده ساختاریافته . . . . . structured metadata
فضای طراحی . . . . . design space
قابل موازی‌سازی . . . . . parallelizable subtasks
قطعات . . . . . chunks
قیود منابع . . . . . resource constraints
مجموعه متون . . . . . corpus
محیط . . . . . environment
مخازن داده‌مجموعه و مدل . . . . . Dataset and Model Repositories
مخازن کد . . . . . code repositories
مدل پایه . . . . . backbone
مدل‌سازی جانشین . . . . . surrogate modeling

مدل‌های زبانی بزرگ Large Language Models (LLMs)
مدیر عامل Agent Manager
مردمی‌سازی democratization
معناداری معنایی semantic meaningfulness
مقیاس scale
منتقد critic
منجمد frozen
منسجم coherent
مهندسی دستور Prompt Engineering
مهندسی ویژگی feature engineering
مولفه‌های ماژولار modular components
میان‌گونه‌ای across modalities
ناهمخوانی inconsistencies
نمایش‌های برداری vector representations
نمایش‌های ساخت‌یافته کلید—مقدار structured key-value representations
نمایش‌های گراف graph representations
نمونه‌برداری نامزد candidate sampling
نهفتارسازی embed
نهفتگی‌ها embeddings
هزینه محاسباتی computational cost
هماهنگی سلسله‌مراتبی Hierarchical Coordination
هماهنگ‌کننده coordinator
هماهنگ‌کننده orchestrator
همریختی isomorphism
همکارانه collaborative
همکاری بین‌عاملی inter-agent collaboration
همکاری مبتنی بر نقش Role-Based Collaboration
هم‌بندی ensemble
وابستگی‌های دوربرد long-range dependencies
وجه‌های داده modalities
پاسخ‌های ایستا static responses
پایداری robustness
پایگاه داده برداری vector database
پایگاه دانش knowledge base

پایگاه‌های داده برداری vector databases
پایگاه‌های دانش knowledge bases
پروتکل‌های ارتباطی communication protocols
پنجره زمینه context window
پیش‌آموزش pre-training
پیش‌پردازش داده data preprocessing
پیکربندی configuration
چارچوب نظری paradigm
چندسطوحی/چندوفایی multi-fidelity
چندعاملی Multi-Agent
کارایی performance
کارایی محاسباتی computational efficiency
کارت‌های داده data cards
کارت‌های مدل model cards
کارگزاران خوانش تخصصی specialized reader agents
کدگذاری رشته‌ای جداکننده‌محور delimited string encodings
کم‌نمونه few-shot
کنترل‌گرهای جریان‌کار workflow controllers
کنشگرها actuators
کیفیت—تنوع quality-diversity
گذارهای بین قالب‌ها format transitions
گراف جهت‌دار بدون‌دور Directed Acyclic Graph (DAG)
گراف محاسباتی computational graph
گزارش وقایع log
گزیننده‌های ویژگی feature selectors
گسسته‌سازی discretized
گفت‌وگوهای چت chat-style dialogues
گونه خروجی output modalities
یادگیری تقویتی Reinforcement Learning
یادگیری درون‌متنی از تاریخچه بهینه‌سازی In-Context Learning from Optimization History
یادگیری زمینه‌ای In-Context Learning (ICL)
یادگیری ماشین خودکار Automated Machine Learning (AutoML)

Abstract:

Traditional AutoML approaches using Bayesian optimization and evolutionary algorithms lack interpretability and struggle to leverage domain knowledge. Large Language Models (LLMs) with capabilities in natural language understanding, reasoning, and code generation offer a transformative paradigm shift. This seminar explores how LLM-based agents can revolutionize AutoML systems by acting as intelligent, context-aware, and adaptive agents rather than blind optimizers. This work systematically reviews recent LLM-based AutoML approaches from three perspectives: (1) Agent Architecture: single-agent systems (direct optimization, evolutionary operators, workflow controllers) versus multi-agent systems (role-based collaboration, hierarchical coordination) (2) Knowledge Sources: internal knowledge (optimization history, in-context learning) versus external retrieval (literature extraction, model repositories) and (3) Output Formats: structured configurations, code generation, tree representations, and hybrid approaches. Analysis reveals an even split between single-agent and multi-agent architectures, with most systems relying on internal knowledge and few leveraging external knowledge through retrieval-augmented generation. Output formats show high diversity, with hybrid approaches balancing machine-readability and expressiveness. The seminar identifies critical open problems and future directions, including resource-constrained optimization, advanced knowledge integration, and multi-agent frameworks. A thesis proposal is presented for designing an agent-based AutoML system for model selection and fine-tuning in transfer learning with limited data.

Keywords: Automated Machine Learning, Large Language Models, Agent-Based Systems,Hyperparameter Optimization, Neural Architecture Search, Retrieval-Augmented Generation, In-Context Learning, Transfer Learning

Iran University of Science and Technology
Computer Engineering Department

Review of agent-based large language models in neural architecture search and hyperparameter optimization

A Seminar Report Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science in Computer Engineering-Artificial Intelligence

By:

Mohammad Sadegh Poulaei Moziraji

Supervisors:

Dr. MohammadReza Mohammadi and Dr. Sauleh Etemadi

November 2025