Journal of Applied And Basic Machine Intelligence Research (ABMIR)

پژوهش‌های نظری و کاربردی هوش ماشینی

سال دوم، شماره اول، بهار و تابستان ۱۴۰۳، صفحات ۹۱ - ۷۷

جستجوی معماری عصبی فشرده برای طبقه‌بندی تصاویر با استفاده از الگوریتم جستجوی گرانشی

سجاد بسطامی، محمدباقر دولتشاهی*

گروه مهندسی کامپیوتر، دانشکده فنی و مهندسی، دانشگاه لرستان، خرم‌آباد، ایران

مقاله پژوهشی

تاریخ دریافت:
۱۴۰۳/۰۷/۱۳

تاریخ پذیرش:
۱۴۰۳/۱۰/۱۱

کلیدواژه‌ها:
الگوریتم جستجوی گرانشی، جستجوی معماری عصبی، شبکه‌های عصبی کانولوشنی، یادگیری عمیق

نویسنده‌مسئول:
dowlatshahi.mb@lu.ac.ir

چکیده

در این مقاله، یک روش جستجوی معماری عصبی فشرده برای طبقه‌بندی تصاویر با استفاده از الگوریتم جستجوی گرانشی(GSA) ارائه‌شده است. یادگیری عمیق با بهره‌گیری از مدل‌های محاسباتی چندلایه، امکان استخراج خودکار ویژگی‌ها را از داده‌های خام در سطوح انتزاعی مختلف فراهم می‌کند که نقش کلیدی در مسائل پیچیده‌ای مانند طبقه‌بندی تصاویر دارد. روش جستجوی معماری عصبی (NAS) که به‌طور خودکار به کشف معماری‌های جدید شبکه‌های عصبی کانولوشنی (CNN) می‌پردازد، با چالش‌هایی نظیر پیچیدگی محاسباتی و هزینه‌های بالا مواجه است. برای مقابله با این چالش‌ها، رویکردی بر پایه الگوریتم جستجوی گرانشی (GSA) توسعه داده‌شده است که بهینه‌سازی دوسطحی با طول متغیر را برای طراحی معماری‌های میکرو و ماکرو شبکه‌های عصبی کانولوشنی (CNN) به کار می‌گیرد. این رویکرد با استفاده از فضای جستجوی فشرده و گلوگاه‌های کانولوشنی اصلاح‌شده، عملکرد بهتری نسبت به روش‌های پیشرفته نشان می‌دهد. نتایج تجربی بر روی مجموعه داده‌های CIFAR-10، CIFAR-100 وImageNet نشان می‌دهد که روش پیشنهادی با دقت طبقه‌بندی 98.48% و هزینه جستجوی 1.05 (روزGPU) از الگوریتم‌های موجود ازنظر دقت، هزینه جستجو و پیچیدگی معماری برتری دارد.

: 10.22034/abmir.2024.22228.1066

E-ISSN: 2821-2037

۱- مقدمه

شبکه عصبی کانولوشن[1] (CNN) یکی از مهم‌ترین شبکه‌های یادگیری ماشین است. مزیت اصلی CNN این است که به‌طور خودکار ویژگی‌های مهم را بدون هیچ نظارت انسانی تشخیص می‌دهد [۱]. CNN ویژگی‌های تصویر مبتنی بر داده‌های قابل یادگیری، یک بازنمایی خوب و سلسله مراتبی را از داده‌های آموزشی استخراج می‌کند[۲]. معماری CNN عملکرد مناسبی را در مسائل بینایی کامپیوتر و یادگیری ماشین نشان داده است. CNN در یک سطح انتزاعی آموزش و پیش‌بینی می‌کند و جزئیات را برای بخش‌های بعدی حذف می‌کند[۳]. طراحی معماری‌های CNN فرآیندی پیچیده و پرزحمت است که به عوامل زیادی در این طراحی نیاز دارد.

معماری آن شامل نوع و تعداد هسته‌ها در هر لایه است و طراحی آن به‌دقت و توجه قابل‌توجهی نیاز دارد. علاوه بر این، CNN ها اغلب بر روی مجموعه داده‌هایی با ویژگی‌های خاص آموزش داده می‌شوند، این داده‌ها ممکن است شامل مجموعه‌ای محدود و تخصصی از تصاویر یا داده‌هایی با ساختار و ویژگی‌های مشخص باشند که به‌طور مستقیم قابل‌تعمیم به سایر برنامه‌ها نیستند و طراحی مدل در لایه‌های عمیق‌تر به‌سختی به‌طور مستقیم بین برنامه‌های مختلف به اشتراک گذاشته می‌شود؛ بنابراین، یک روش کارآمد برای خودکار کردن فرآیند طراحی معماری CNN موردنیاز است. جستجوی معماری عصبی[2] (NAS)را می‌توان به‌عنوان زیرشاخه یادگیری ماشین خودکار در نظر گرفت و با بهینه‌سازی هایپر پارامترها و انواع یادگیری، انعطاف‌پذیری قابل‌توجهی ایجاد کرد. روش‌های NAS به سه فاز تقسیم می‌شوند: فضای جستجو، استراتژی جستجو و استراتژی تخمین عملکرد.

هدف اصلی این روش، کاهش حداکثری دخالت انسانی و طراحی خودکار معماری‌های عصبی بهینه توسط الگوریتم است. هدف NAS طراحی معماری عصبی است که با استفاده از منابع محاسباتی محدود و به‌کارگیری روش‌های خودکار، با کاهش حداکثری دخالت انسانی به بهترین عملکرد ممکن دست می‌یابد[٤]. یک مسئله عمده در NAS ناشی از طراحی فضای جستجو این است که برای طراحی گلوگاه معماری از روش یادگیری سنتی استفاده می‌کنند و عملکرد معماری تا حدودی کاهش می‌یابد[۵]. در مورد روش‌های جستجو،NAS قدیمی اغلب مبتنی بر یادگیری تقویتی برای کشف معماری‌های با عملکرد خوب هستند. اغلب از هزینه‌های منابع محاسباتی رنج می‌برند که اغلب به دلیل بهینه‌سازی غیرقابل‌حل و همگرایی کند جستجوی معماری است.

روش‌های مبتنی بر گرادیان کارآمدتر هستند اما باوجود هزینه‌های بالای حافظه GPU، استفاده از روش‌های بهینه‌سازی سراسری برای جلوگیری از گیر افتادن در مینیمم‌های محلی ضروری به نظر می‌رسد. الگوریتم جستجوی گرانشی[3] (GSA) یک تکنیک بالقوه با تعدادی از مزایای کلیدی است که در حل بسیاری از مسائل پیچیده بهینه‌سازی با موفقیت استفاده‌شده است. یکی از مزایای کلیدی GSA قابلیت تطبیق با تغییرات توابع هدف و مدیریت مناسب فضای جستجو، امکان بهینه‌سازی در مسائل چندبعدی و پیچیده را با دقت و کارایی بالا فراهم می‌آورد. حل برخی از مسائل پیچیده بهینه‌سازی با استفاده از تکنیک‌های سنتی با محدودیت‌هایی مواجه است و نیاز به روش‌های پیشرفته‌تری دارد[٦]. GSA با اجتناب از حداقل‌های محلی و سرعت همگرایی، عملکرد بهتری را در مقایسه با روش‌های فراابتکاری مانند ACO و PSO نشان داده است[۷] پارامترهای کنترلی GSA ازجمله ثابت گرانش، عامل وزن با بسیاری از سناریوهای جستجو سازگار است و حساسیت کمتر نسبت به شرایط اولیه و همچنین تغییرات توابع هدف دارد [۸].

الگوریتم‌های NAS معمولاً با تعداد زیادی پارامتر برای ارزیابی عملکرد معماری‌های کاندید مواجه هستند. ارزیابی هر معماری غالباً به صدها دوره آموزشی مبتنی بر گرادیان بر روی مجموعه داده‌های آموزشی نیاز دارد که این امر هزینه محاسباتی بسیار بالایی را به همراه دارد. ازنظر معماری، بسیاری از کارهای قبلی برای

[1] Convolutional Neural Network

[2] Neural Architecture Search
[3] Gravitational Search Algorithm

بهبود عملکرد منجر به طراحی معماری‌هایی با ساختارهای پیچیده و اندازه‌های بزرگ‌شده است. ازاین‌رو، طراحی معماری‌های فشرده و بهینه که بتوانند ضمن کاهش پیچیدگی محاسباتی، عملکرد مطلوبی ارائه دهد ضروری است. روش‌های فراابتکاری این استراتژی‌ها با استفاده از فرآیندهای اکتشافی و تکراری، فضای مسئله را جستجو کرده و راه‌حل‌های مناسبی ارائه می‌دهند[۹] .

الگوریتم جستجوی گرانشی با رویکرد مبتنی بر جمعیت، از چندین راه‌حل اولیه تصادفی شروع می‌شود. سپس، یک الگوریتم فرآیند تکراری را برای دستیابی به یک‌راه حل نزدیک به بهینه انجام می‌دهد. در طول تکرارها، جمعیت به‌تدریج با استفاده از عملیات الهام گرفته از ماهیت فیزیک و طبیعت به سمت راه‌حل‌های بهتر همگرا می‌شوند. الگوریتم از یک جمعیت تصادفی شروع می‌شود و یک حلقه را اجرا می‌کند که شامل ارزیابی عوامل و به‌روزرسانی جمعیت تا زمانی که معیارهای توقف برآورده شود. روش به‌روزرسانی با استفاده از عامل‌های الگوریتم که به‌طور خلاقانه با تقلید از طبیعت طراحی‌شده‌اند، انجام می‌شود. عوامل جدید با استفاده از ترکیب عوامل فعلی تولید می‌شوند. عامل‌ها بعد از یافتن فضای جستجو به سمت نقاط جدید حرکت می‌کنند. الگوریتم جستجوی گرانشی، توسط مفاهیم گرانش تعریف می‌شود. میدان گرانشی به‌صورت منحنی فضا–زمان ظاهر می‌شود که توسط نظریه نسبیت عمومی اینشتین توصیف‌شده است [۱۰]؛ بنابراین، یک جهش بزرگ در این زمینه برای اتخاذ مفاهیم گرانش در تولید عوامل جستجوی کارآمد وجود دارد.

بر این اساس، GSA را می‌توان به‌عنوان یک الگوریتم جستجوی فراابتکاری مبتنی بر جمعیت و مبتنی بر فیزیک در نظر گرفت. ازآنجاکه جرم و فاصله بر نیروی گرانشی تأثیر می‌گذارند، عوامل از طریق گرانش همکاری و رقابت می‌کنند. برهم‌نهی نیروهای گرانشی، وابستگی به فاصله و ارتباط بین مقادیر جرم و برازندگی[1] این الگوریتم را منحصربه‌فرد می‌کند[۹]. با توجه به محدودیت‌های NAS، این مقاله باهدف توسعه یک الگوریتم GSA انعطاف‌پذیر برای طراحی معماری CNN کارآمد و فشرده است. به‌طور خاص، یک رویکرد GSA طول متغیر دوسطحی برای تکامل معماری میکرو و معماری ماکرو شبکه CNN طراحی‌شده است. نوآوری‌های این مقاله برای NAS فشرده عبارت‌اند از:

(۱) فرمول سازی یک تابع هدف برای بهینه‌سازی، شامل همه مفروضات و محدودیت‌ها، از مسئله فضای جستجو و چارچوب احتمالی.

(۲) استفاده از یک بلوک کانولوشنی گلوگاه معکوس متحرک برای ایجاد فضای جستجو.

(۳) توسعه GSA دوسطحی با کدگذاری لایه‌های کانولوشنی باقابلیت تغییر فضای جستجوی جمعیت برای پیکربندی معماری CNN.

(٤) نمایش GSA دوسطحی و طول متغیر با یک شتاب یکپارچه برای تأیید عملکرد بهتر آن به‌منظور کاهش تعداد فیلترها و کاهش وضوح ویژگی‌ها پیشنهاد می‌شود که موجب افزایش سرعت همگرایی فضای جستجو و کاهش هزینه‌های جستجو در مجموعه داده‌ها می‌گردد.

از نتایج تجربی بر روی سه مجموعه داده cifar-10،cifar100، imagenet [۱۱] که برای طبقه‌بندی تصویر استفاده می‌شود و همچنین مقایسه با رقبای پیشرفته، نشان می‌دهیم که در بین تمام روش‌های مقایسه شده، روش پیشنهادی فشرده‌ترین معماری را با عملکرد بسیار خوب و با پیچیدگی محاسباتی کم‌تر استخراج می‌کند. بخش ۲ مراحل تدوین تابع هدف و کارهای مرتبط را توضیح می‌دهد. بخش ۳ در مورد GSA با طول متغیر کارآمد را ارائه می‌دهد. بخش ٤ نتایج آزمایش‌ها را ارائه می‌دهد. در بخش ۵، نتایج جمع‌بندی شده و مقاله با ارائه نتیجه‌گیری نهایی خاتمه می‌یابد.

۲- کارهای مرتبط

مسئله جستجوی معماری عصبی با در نظر گرفتن چالش‌های مرتبط با مدل‌سازی هدف و طراحی بهینه، بر اساس تحقیقات پیشین فرمول‌بندی شده است.

[1] fitness

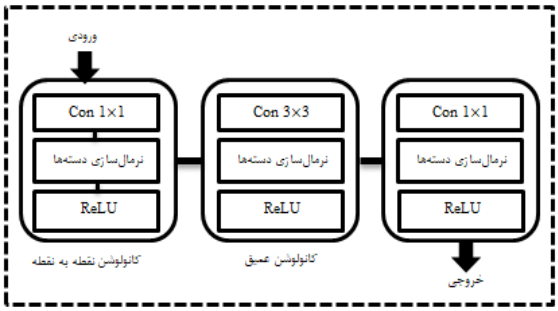
۲-۱ موبایل نت[1]

یک کلاس از مدل‌های کارآمد برای موبایل نت ارائه می‌کند[۱۲]. این کلاس از مدل‌ها بهینه‌سازی شده‌اند تا مصرف منابع محاسباتی و انرژی را کاهش داده و به‌ویژه برای دستگاه‌های با توان پردازشی محدود، مانند گوشی‌های هوشمند، مناسب باشند. موبایل نت مبتنی بر معماری ساده‌ای است که از کانولوشن برای ساخت شبکه‌های عصبی عمیق سبک‌وزن استفاده می‌کند. دو متا پارامتر سراسری را معرفی می‌کند که به‌طور کارآمد خطا و دقت را بررسی می‌کند. متا پارامترها مقادیر تنظیم‌پذیری هستند که خارج از فرآیند یادگیری تعیین می‌شوند و بر عملکرد مدل تأثیر می‌گذارند، مانند نرخ یادگیری یا اندازه فیلترها. مدل موبایل نت بر اساس لایه کانولوشن عمیق که دارای یک کانولوشن استاندارد و یک کانولوشن ۱×۱ است. برای موبایل نت، کانولوشن عمیق یک فیلتر را برای هر کانال ورودی اعمال می‌کند.

یک کانولوشن استاندارد ورودی‌ها را فیلتر کرده و آن‌ها در یک مجموعه جدید از خروجی‌ها ترکیب می‌شوند. کانولوشن عمیق این مجموعه را به دولایه تقسیم می‌کند: یک‌لایه جداگانه برای فیلتر کردن و یک‌لایه جداگانه برای ترکیب. این فرآیند تأثیر زیادی بر کاهش محاسبات و اندازه مدل دارد. لایه کانولوشن استاندارد یک نگاشت ویژگی $\boldsymbol{df \times df \times m}$ را به‌عنوان ورودی f می‌گیرد و یک نگاشت ویژگی $\boldsymbol{df \times df \times n}$ تولید می‌کند که در آن df عرض و ارتفاع فضایی یک ویژگی با ورودی مربعی است، m تعداد کانال‌های ورودی است (عمق ورودی)، df عرض و ارتفاع فضایی نگاشت ویژگی خروجی و n تعداد کانال خروجی (عمق خروجی) است[۱۳]. معماری موبایل نت در شکل ۱ نشان داده‌شده است. اندازه پارامترها از طریق فرمول (۱) کاهش می‌یابد:

$$R = \frac{df^2 m + mn}{df^2 mn} = \frac{1}{2} + \frac{1}{df^2} \qquad (۱)$$

R کاهش پارامترها، df عرض و ارتفاع فضایی ویژگی‌ها، m تعداد کانال‌های ورودی، n تعداد کانال خروجی است.



شکل (۱): معماری موبایل نت

۲-۲ طراحی معماری فشرده خودکار CNN

یک الگوریتم تکاملی برای NAS چند هدفه به نام LEMONADE که توابع پیش‌بینی‌کننده و محدودیت‌های منابع را بهینه می‌کند. NAS به‌عنوان یک مسئله چندهدفه دو موضوع را مینیمم می‌کند: خطای طبقه‌بندی و تعداد عملیات ممیز شناور[۱۴]. Mnasnet خطای مدل را در تابع هدف NAS محاسبه کرده و یک فضای جستجوی سلسله مراتبی توسط بلوک‌های کانولوشنی را بررسی می‌کند که منجر به معماری‌های بسیار کارآمد می‌شود. از فضای جستجوی مبتنی بر الگوی موبایل نت استفاده می‌کند و تابع هدف دارای پیچیدگی محاسباتی را برای بهبود کارایی جستجو مشخص می‌کند[۱۵]. پیچیدگی معماری را می‌توان با تعداد بلوک‌ها و تعداد گره‌ها در هر بلوک اندازه‌گیری کرد و از یک الگوریتم تکاملی دوسطحی برای طراحی معماری فشرده CNN استفاده کرد.

یک روش NAS مبتنی بر GA[2] برای طراحی معماری CNN پیشنهادشده است[۱۶]. یک بلوک کانولوشنی اصلاح‌شده در فضای جستجو اضافه‌شده است که طراحی CNN را آسان می‌کند. بسیاری از روش‌های دیگر شامل محدودیت‌هایی هستند که ممکن است بر انعطاف‌پذیری الگوریتم‌های جستجو تأثیر بگذارد. این کار بر بهینه‌سازی عملکرد طبقه‌بندی و تعداد پارامترها تمرکز دارد. یک فضای جستجو از معماری سبک‌تر ساخته‌شده است که شامل بلوک‌های کانولوشنی با پارامترهای اصلاح‌شده و اتصالات پراکنده و فیلترهای کمتر است. روش پیشنهادی می‌تواند چنین فضای جستجویی را با انعطاف‌پذیری بیشتری برای طراحی معماری‌های فشرده بدون پیچیدگی‌های محاسباتی استخراج نمود.

[1] MobileNets

[2] Genetic Algorithm

۳- الگوریتم جستجوی گرانشی برای طراحی معماری فشرده CNN

ازآنجاکه مسئله جستجوی تابع هدف NP سخت[1] است[۱۳] [۱۲]، زمان محاسبه با افزایش ابعاد مسئله برای یافتن راه‌حل بهینه بسیار افزایش می‌یابد و غیرقابل‌حل می‌شود. بنابراین، یک رویکرد اکتشافی مانند GSA می‌تواند گزینه خوبی برای حل مسئله جستجوی بهینه مانند این مطالعه باشد.

۳-۱ الگوریتم جستجوی گرانشی

گرانش و جرم[2] مفاهیم اصلی GSA هستند[۶]. در این الگوریتم، عوامل جستجو به‌عنوان اجسامی با جرم خاص در نظر گرفته می‌شوند که از طریق نیروی گرانش با یکدیگر تعامل دارند. موقعیت هر عامل نشان‌دهنده یک راه‌حل پیشنهادی برای مسئله است و جرم هر عامل[3] بر اساس یک تابع هدف محاسبه می‌شود. نیروی گرانشی به‌طور هم‌زمان باعث می‌شود که همه اجسام به سمت راه‌حل‌های بهینه حرکت کنند [۹]. در GSA، عوامل با استفاده از جرم‌ها و نیروهای متقابل ارزیابی می‌شوند و هر ذره بر اساس یک تابع هزینه که کیفیت راه‌حل‌ها را مشخص می‌کند، سنجیده می‌شود [۱۷].مجموعه‌ای از N راه‌حل تصادفی با یک بردار-d بعدی به شرح زیر تولید می‌شود:

$$i = 1, \dots, N, m_i(t) = \{m_i^1, \dots, m_N^q\} \quad (۲)$$

نیروهای کششی از جرم i به جرم j در متغیر h $(h \in 1,\dots,d)$ در زمان خاص t، به‌صورت زیر محاسبه می‌شود:

$$F_{ij}^h(t) = G(t)\frac{M_i(t).M_j(t)}{R_{ij}(t)+\varepsilon}\left(m_j^h(t) - m_j^h(t)\right) \quad (۳)$$

در معادله (۳) G(t) نشان‌دهنده ثابت گرانش است، درحالی‌که M_i و M_j به ترتیب نیروهای گرانشی جرم i و j هستند، R_{ij} فاصله اقلیدسی بین عناصر i و j است، و ε یک مقیاس کوچک است زمانی که $R_{ij} \approx 0$ باشد از نتایج ناپایدار جلوگیری می کند. $G(t) = G(t_0)e^{-\propto.t/T}$ که $G(t_0)$ و $\propto$ معمولاً به ترتیب ۱۰۰ و ۲۰ است [۶].نیروی نهایی که بر روی راه‌حل کاندیدا i عمل می‌کند به شرح زیر است:

$$F_i^h(t) = \sum_{j=1,j\neq i}^{N} \text{rand}.F_{ij}^h(t) \quad (۴)$$

rand یک عدد تصادفی معمولاً توزیع‌شده در محدوده [۰،۱] است، بنابراین، شتاب راه‌حل i در جهت h ام به‌صورت زیر محاسبه می‌شود:

$$a_i^h(t) = \frac{F_i^h(t)}{M_{ii}(t)} \quad (۵)$$

M_{ii} *جرم اینرسی عامل* i*ام‌است. سپس، سرعت*[4] *و موقعیت جدید کاندید* i *را می‌توان به شرح زیر تعیین کرد:*

$$\begin{cases} v_i^h(t+1) = rand.v_i^h(t) + a_i^h(t) \\ m_i^h(t+1) = m_i^h(t) + v_i^h(t) \end{cases} \quad (۶)$$

کیفیت هر راه‌حل با یک تابع هزینه[5] ارزیابی می‌شود، از چنین مقادیر برازندگی برای تعیین جرم‌های گرانشی جدید در هر تکرار استفاده می‌شود که در زیر نشان داده‌شده است:

$$P_i(t) = \frac{f(m_i(t)) - worst(t)}{best(t) - worst(t)} \quad (۷)$$

$$M_i(t) = \frac{P_i(t)}{\sum_{i=1}^{N} P_i(t)} \quad (۸)$$

که در آن تابع هزینه به‌صورت f(0)، worst(t) و best(t) بدترین بهترین راه‌حل تابع هزینه هستند.

۳-۲ ساختار الگوریتم پیشنهادی

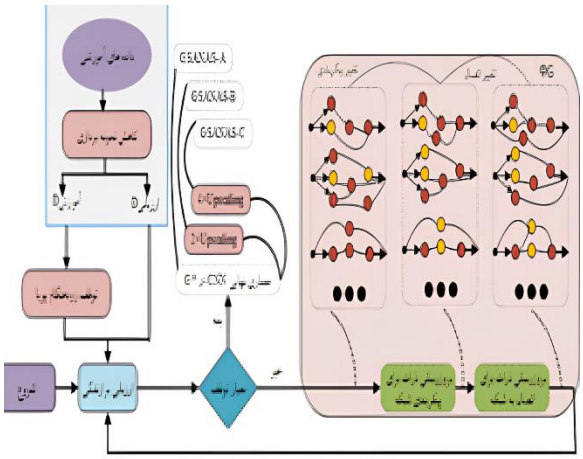
هدف NAS این است که به‌طور خودکار معماری بهینه شبکه G^* را در یک فضای جستجوی از پیش تعریف‌شده φ_Gکشف کند، که پس از آموزش، بتواند کمترین خطا را در مجموعه ارزیابی $D_{\text{ارزیابی}}$ به دست آورد. این فرآیند را می‌توان به‌صورت زیر فرمول‌بندی کرد:

$$\begin{cases} G^* = arg\ min_{G\in\varphi_G} L(G, w_G^*, D_{\text{ارزیابی}}) \\ s.t. \quad w_G^* = arg\ min_{w_G} L(G, w_G, D_{\text{آموزش}}) \end{cases} \quad (۹)$$

که در آن $L(G, w_G, D_{\text{آموزش}})$ عملکرد معماری شبکه G با وزن‌های w_G بر روی‌داده‌های D را اندازه‌گیری می‌کند و w_G^* نشان‌دهنده وزن‌های معماری بهینه شبکه G^* است که بهترین

[1] NP-hard
[2] mass
[3] agent
[4] velocity
[5] cost function

عملکرد را بر روی‌داده‌های آموزشی $D_{\text{آموزش}}$ به دست می‌آورد. در این کار، w_Gبه‌عنوان یک فضای جستجو توسط بلوک کانولوشنی طراحی‌شده است. G^* توسط GSA با طول متغیر دوسطحی در بهینه‌سازی سطح بالا بر اساس وزن‌های w_G^* که در بهینه‌سازی سطح پایین یاد گرفته‌شده است، جستجو می‌شود. چارچوب کلی روش پیشنهادی از جریان الگوریتم NAS مبتنی بر معیار توقف زودهنگام معمولی پیروی می‌کند که در شکل ۲ نشان داده‌شده است.



شکل (۲): چارچوب الگوریتم پیشنهادی GSA برای جستجوی NAS. یک‌ذره در این معماری از دو زیرذره تشکیل‌شده است، یک زیرذره برای پیکربندی شبکه (sp-conf) و زیرذره دیگر برای اتصال شبکه (sp-conn).

روش پیشنهادی با تولید تصادفی ذره‌ها آغاز می‌شود که به‌عنوان یک جمعیت شناخته می‌شوند. به دلیل بزرگی فضای جستجو، جمعیت اولیه باید تنوع زیادی داشته باشد تا فرآیند تکاملی به‌خوبی عمل کند. بخش اصلی الگوریتم شامل یک فرآیند تکاملی است که از یک حلقه ارزیابی برازندگی و به روزرسانی دوسطحی GSA تشکیل‌شده است. ذره‌ای که معماری CNN را نمایش می‌دهد از دو سطح تشکیل می‌شود: سطح میکرو و سطح ماکرو. زیرذره سطح میکرو که پیکربندی هر لایه را رمزگذاری می‌کند و به عنوان sp-conf شناخته می‌شود، با استفاده از یک طرح تکاملی GSA پیوسته و با طول متغیر به‌روزرسانی می‌شود. زیرذره دیگر که اتصالات بین لایه‌ها را رمزگذاری می‌کند و به‌عنوان sp-conn شناخته می‌شود، با روش به‌روزرسانی GSA باینری تکامل می‌یابد. در این مقاله از الگوریتم جستجوی گرانشی برای طراحی خودکار معماری‌های فشرده شبکه‌های عصبی کانولوشنی استفاده‌شده است. این الگوریتم به دلیل قابلیت‌های منحصربه‌فرد خود در مدیریت فضای جستجوی پیچیده، اجتناب از مینیمم‌های محلی و انعطاف‌پذیری بالا برای مسائل بهینه‌سازی چندبعدی انتخاب‌شده است. هدف اصلی این روش، کاهش پیچیدگی معماری و هزینه محاسباتی در کنار حفظ یا بهبود دقت مدل‌های CNN است.

درروش پیشنهادی، ابتدا یک مجموعه داده آموزشی کوچک‌تر با کاهش نمونه‌گیری از مجموعه اصلی ایجاد می‌شود. این مجموعه به دو بخش برای جستجوی معماری و ارزیابی تقسیم می‌گردد. در هر مرحله از حلقه تکاملی، ابتدا sp-conf ها و سپس sp-conn ها به‌روزرسانی می‌شوند. هر معماری جدید شبکه عصبی کانولوشنی (CNN) که تکامل‌یافته است، با استفاده از روش گرادیان کاهشی تصادفی[1] (SGD) و یک سیاست توقف زودهنگام مناسب، بر روی بخش مجموعه کاهش‌یافته آموزش داده می‌شود و سپس بر روی بخش ارزیابی، برای اندازه‌گیری برازندگی آن‌ها (مثل دقت طبقه‌بندی)، بررسی می‌شود. درنهایت، وقتی معیار توقف (مثل تعداد مشخصی از تکرارهای جستجوی معماری) برآورده شد، بهترین معماری CNN انتخاب می‌شود. این معماری بر روی کل مجموعه آموزشی و تست به همراه دو نسخه بزرگ‌تر (با افزایش تعداد فیلترهای لایه‌های کانولوشن) آموزش داده می‌شود تا عملکرد نهایی آن‌ها بررسی گردد.

۳-۳ طراحی فضای جستجو

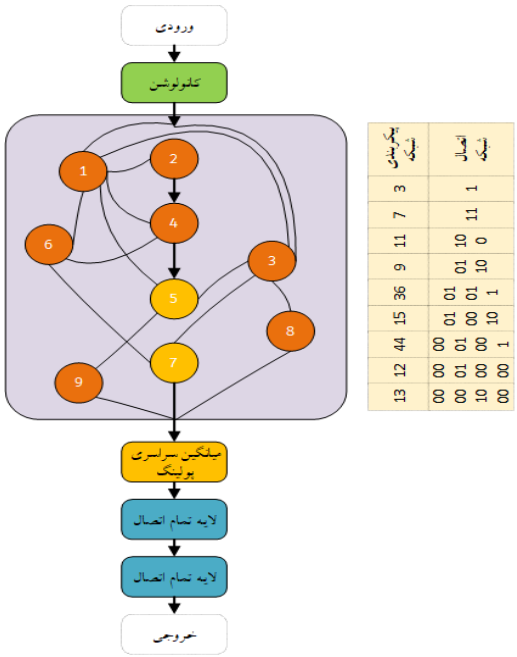
یک معماری معمولی شبکه عصبی کانولوشنی (CNN) شامل سه نوع لایه است: لایه کانولوشنی، لایه تجمعی و لایه کاملاً متصل [۱۸].لایه‌های کانولوشنی و تجمعی[2] که وظیفه کاهش ابعاد فضایی ویژگی‌ها (مانند ارتفاع و عرض) را بر عهده دارد، به‌صورت پشته‌ای قرار می‌گیرند تا ویژگی‌ها را به‌طور کارآمد از داده‌های ورودی استخراج کنند، درحالی‌که لایه‌های کاملاً متصل در انتهای شبکه برای انجام طبقه‌بندی استفاده می‌شوند. در این مطالعه، ما کل بخش کانولوشنی (شبکه استخراج ویژگی) را در CNN برای طبقه‌بندی

[1] stochastic gradient descent

[2] Pooling Layer

تصاویر تکامل می‌دهیم. همان‌طور که در شکل ۳ مشاهده می‌شود، معماری پیشنهادی CNN از یک ساختار تکامل‌یافته استفاده می‌کند. این معماری با یک‌لایه کانولوشن ابتدایی آغاز می‌شود و با یک‌لایه پولینگ میانگین سراسری[1] (GAP) و دولایه متصل کامل به پایان می‌رسد. بخش میانی شبکه، شامل یک ماژول تکاملی با ۹ گره محاسباتی است.

هر یک از این گره‌ها پیکربندی مختلفی از عملیات محاسباتی ازجمله لایه‌های میانگین‌گیری (در گره‌های ۵ و ۷) و بلوک‌های MBConv اصلاح‌شده[2] را انجام می‌دهد، ماژول MBConv [۱۹] یک پارادایم محاسباتی بهینه با عملکرد بالا است که نسبت به ماژول‌های کانولوشن معمولی، پیچیدگی محاسباتی کمتری دارد. ساختار اتصال انعطاف‌پذیر این ماژول، امکان ایجاد طیف گسترده‌ای از معماری‌های شبکه عصبی را فراهم می‌کند، ازجمله معماری‌هایی که شباهت زیادی به ResNet [۲۰] و DenseNet [۲۰] دارند. در این ساختار، هر گره می‌تواند ترکیبی از خروجی‌های گره‌های قبلی را به‌عنوان ورودی خود دریافت کند. گره‌هایی که هیچ اتصال خروجی ندارند، به‌عنوان خروجی نهایی استخراج‌کننده ویژگی تکامل می‌یابند و به طبقه‌بند ارسال می‌شوند. به‌عنوان‌مثال، در شکل ۳، خروجی ترکیبی از گره‌های ۷، ۸ و ۹ به‌عنوان ورودی طبقه‌بند استفاده می‌شود. معماری‌های مطلوب CNN باید فشرده و با تعداد پارامترهای کم باشند. برای دستیابی به این هدف، از بلوک‌های اصلاح‌شده MBConv با کارایی بالا استفاده می‌کنیم که جستجوی معماری‌های فشرده را ساده‌تر می‌کنند. در این طراحی، اتصال میانی MobileNetV3 حذف‌شده و بلوک فشرده[3] (SE) با ماژول توجه کانال کارآمد[4] (ECA) جایگزین شده است که به‌صورت کارآمد تعامل بین کانال‌ها را با یک کانولوشن ۱ بعدی انجام می‌دهد. شکل ٤ نشان می‌دهد بلوک MBConv اصلاح‌شده شامل یک کانولوشن ۱×۱، یک کانولوشن عمیق $d \times d$ و یک ماژول ECA برای یادگیری توجه کانال‌ها است.



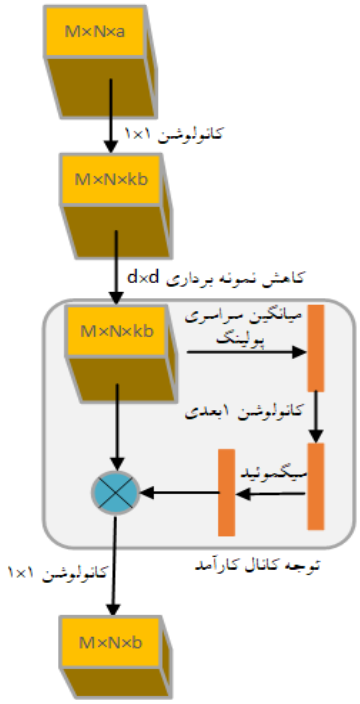
شکل (۳): معماری CNN تکامل‌یافته با استفاده از GSA

درنهایت، خروجی از طریق یک کانولوشن ۱×۱ دیگر دریافت می‌شود. این بلوک با توجه به تعداد فیلترهای خروجی b، اندازه کرنل d و نسبت گسترش k پیکربندی می‌شود. ازآنجاکه انتخاب‌های محدودی برای d و k وجود دارد، تنظیم تعداد فیلترها b به‌عنوان پارامتر اصلی در نظر گرفته می‌شود. طراحی فضای جستجو ویژگی‌های زیر را دارد: اول، به الگوریتم اجازه می‌دهد تا معماری‌های شبکه تعداد متغیر گره‌ها را پیدا کند که عمق مدل و تعداد فیلترهای هر بلوک MBConv را تعیین می‌کند. همچنین تعداد، نوع و مکان لایه‌های pooling که وضوح فضایی معماری را تغییر می‌دهند کدگذاری شده‌اند. گره‌ها به ترتیب هدف‌گذاری می‌شوند و می‌توانند به‌طور دلخواه به گره‌های قبلی متصل شوند که امکان طراحی معماری‌های انعطاف‌پذیر و جدید را فراهم می‌کند. این فضا به GSANAS اجازه می‌دهد تا عمق، عرض، وضوح فضایی و روش‌های اتصال معماری CNN را به‌طور هم‌زمان تکامل دهد.

[1] global average pooling
[2] mobile inverted bottleneck convolution
[3] squeeze-and-excitation
[4] Efficient Channel Attention

۳-٤ ارزیابی برازندگی

در مرحله جستجوی تکراری معماری شبکه، هر ذره جدید به‌روزرسانی شده ارزیابی می‌شود تا میزان برازندگی آن مشخص گردد. ارزیابی معمولاً شامل آموزش کامل معماری CNN رمزگشایی‌شده بر روی مجموعه آموزشی $D_{\text{آموزش}}$ و سپس بررسی عملکرد آن بر روی مجموعه ارزیابی تناسب $D_{\text{ارزیابی}}$ است. به‌هرحال، این فرآیند سهم عمده‌ای از پیچیدگی محاسباتی الگوریتم NAS را دارد.



شکل (٤): بلوک MBConv اصلاح‌شده با ورودی a کانال و خروجی b کانال.

برای کاهش بار محاسباتی در جستجو و ارزیابی معماری، طرحی کارآمد برای سرعت بخشیدن این فرآیند پیشنهاد می‌کنیم که با ترکیب چندین روش کاهش محاسباتی، ازجمله معیار توقف زودهنگام، کاهش نمونه‌برداری تصویر، کاهش مقیاس معماری به دست می‌آید. کد شبه ارزیابی برازندگی در الگوریتم ۱ نمایش داده‌شده است.

۳-۵ به‌روزرسانی ذره

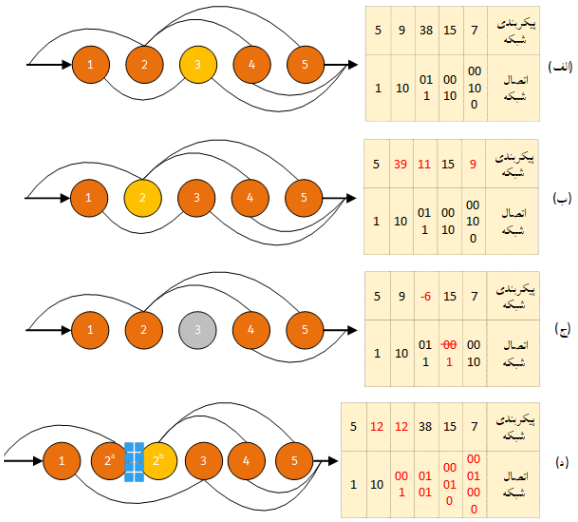
یک معماری CNN بر اساس دو قسمت sp-conf و sp-conn تعریف می‌شود؛ بنابراین، تکامل معماری شبکه با به‌روزرسانی هم‌زمان این دو بخش است که با استفاده از یک الگوریتم GSA دوسطحی انجام می‌شود. به‌روزرسانی sp-conf نه‌تنها به تکامل پیکربندی بلوک‌ها و لایه‌ها می‌پردازد، بلکه ممکن است تغییراتی در طول شبکه ایجاد کند. پس از به‌روزرسانی sp-conf و پیش از به‌روزرسانی sp-conn، درصورتی‌که تغییری در ابعاد sp-conf رخ دهد، sp-conn باید متناسب با آن تنظیم شود.

الگوریتم (۱): شبه کد ارزیابی برازندگی GSANAS

/*مقداردهی اولیه:*/

۱- ایجاد یک جمعیت اولیه p از اشیاء به‌طور تصادفی;
۲- تقسیم مجموعه داده به $D_{\text{آموزش}}$ و $D_{\text{ارزیابی}}$
۳- حلقه foreach ذرات در جمعیت *do*:
٤- ایجاد یک بهینه‌ساز گرادیان با ε دوره;
٥- X_i را به موقعیت ذره اختصاص دهید;
٦- مقدار برازندگی هر ذره را محاسبه کنید;
۷- به‌روزرسانی G، محاسبه M و a برای هر عامل
۸- مقدار *local_best* هر ذره را برای خودتنظیم کنید;
۹- بهترین شتاب هر ذره بر اساس $D_{\text{آموزش}}$قرار دهید;
۱۰- پایان
۱۱- *global_best* را بر روی بهترین ذره مناسب قرار دهید;
/* تکاملات:*/
۱۲- *for k ← 1 to max_generation do*
۱۳- حلقه foreach ذرات در جمعیت *do*:
۱٤- محاسبه سرعت حرکت ΔX^{i+1}; /* معادله (٥)*/
۱٥- محاسبه موقعیت جدید X^{i+1} ; /* معادله (٥)*/
۱٦- برازندگی X_{i+1} را بروز کنید; /*معادله (۷)*/
۱۷- L_{k+1} *local_best* را بروز کنید; /* معادله (۸)*/
۱۸- پایان
۱۹- *global_best* G_{k+1} را بروز کنید; /* معادله (۹)*/
۲۰- پایان

همان‌طور که در شکل ۵ مشاهده‌شده است در شکل ۵ (الف) یک‌ذره و معماری مرتبط با آن را نشان می‌دهد و شکل ۵ (ب) به‌روزرسانی ذره بدون تغییر طول را نشان می‌دهد. اگر در طول فرآیند به‌روزرسانی یک گره حذف شود، اطلاعات رمزگذاری شده مرتبط باید از sp-conn حذف شود، همان‌طور که در شکل ۵ (ج) نشان داده‌شده است. اگر یک گره به دو گره تقسیم شود، این دو گره به هم متصل می‌شوند و اتصالات ورودی و خروجی گره اصلی به آن‌ها اختصاص می‌یابد. این دو روش به‌روزرسانی به‌طور مشترک به جستجوی کلی معماری کمک می‌کنند. پس از فرآیند به‌روزرسانی، هنگام رمزگشایی و ارزیابی معماری جدید، اگر یک گره، ویژگی با وضوح فضایی مختلف را به‌عنوان ورودی دریافت کند، یک عملیات استاندارد کانولوشن برای کاهش اندازه ویژگی با وضوح بالاتر اضافه می‌شود تا اطمینان حاصل شود که اندازه‌های آن‌ها برای ادغام یکسان است.



شکل (۵): به‌روزرسانی sp-conf و تغییرات آن در توپولوژی اتصال. (الف). ذره و معماری قبل از به‌روزرسانی sp-conf(ب). ذره و معماری پس از به‌روزرسانی sp-conf بدون تغییر طول (ج). ذره و معماری پس از به‌روزرسانی sp-conf که در آن‌یک گره حذف‌شده است (د). ذره و معماری پس از به‌روزرسانی sp-conf که در آن‌یک گره به دو گره تقسیم‌شده است.

این تغییر می‌تواند توسط هر دو به‌روزرسانی ایجاد شود، بنابراین مشخص است که پارامتر معماری و تغییرات وضوح فضایی توسط هر دو فرآیند به‌روزرسانی تحت تأثیر قرار می‌گیرد.

٤- آزمایش‌ها

برای ارزیابی عملکرد GSANAS، ما شبیه‌سازی، مقایسه و آزمایش‌هایی به شرح زیر انجام داده‌ایم. سه مجموعه داده طبقه‌بندی تصویر معیار و رقبا انتخاب‌شده موردبحث قرار می‌گیرند.

٤-١ مجموعه داده

برای ارزیابی عملکرد GSANAS، از سه مجموعه داده مرسوم NAS یعنی CIFAR-10 [۲۱]، CIFAR-100 [۲۱]و ImageNet [۲۲] استفاده کرده‌ایم. این مجموعه‌ها در سال‌های اخیر برای طبقه‌بندی مبتنی بر CNN بسیار مورداستفاده قرارگرفته‌اند و چالشی برای مقایسه عملکرد فراهم می‌کنند. CIFAR-10 شامل ۶۰،۰۰۰ (۵۰،۰۰۰ آموزش، ۱۰،۰۰۰ تست) تصویر RGB با اندازه ۳۲×۳۲ است که به ۱۰ کلاس تقسیم‌شده‌اند. CIFAR-100 مشابه CIFAR-10 است اما دارای ۱۰۰ کلاس است، که طبقه‌بندی را چالش‌برانگیزتر می‌کند. برای بررسی بیشتر، GSANAS بر روی ImageNet که دارای ۱،۰۰۰ کلاس و بیش از ۱,۲ میلیون تصویر برای آموزش و ۵۰،۰۰۰ تصویر برای تست با اندازه ۲۲٤×۲۲٤ است نیز آزمایش می‌شود. در آزمایش‌ها، ۱۰٪ از داده‌های آموزشی CIFAR به‌عنوان مجموعه ارزیابی جداشده و باقی‌مانده برای جستجوی معماری استفاده می‌شود. تصاویر آموزشی CIFAR-10 و CIFAR-100 با تکنیک‌هایی مانند zero padding، برش تصادفی و چرخش افقی تصادفی بهبود داده می‌شوند.

الگوریتم GSANAS از الگوریتم GSA دوسطحی برای بهینه‌سازی ساختار شبکه استفاده می‌کند. پارامترهای GSA که شامل ضرایب شناختی، اجتماعی و وزن اینرسی هستند، نقش مهمی در عملکرد الگوریتم دارد. مقادیر این پارامترها در جدول ۱ بر اساس کارهای قبلی [۹،۲۳،۲٤].تعیین‌شده است. همچنین، اندازه جمعیت و تعداد نسل‌ها به ترتیب به ۲۰ تنظیم‌شده‌اند تا به یک تعادل مناسب بین اکتشاف و بهره‌برداری در فضای جستجو دست‌یابیم. علاوه بر این، برای کنترل پیچیدگی شبکه، تعداد

اتصالات بین زیر شبکه‌ها به کمتر از ٦٤ محدودشده است. محیط آزمایشی ما شامل پردازنده Intel® Core™ i7-10750H با فرکانس ۲,٦۰ گیگاهرتز، حافظهDDR4 با ظرفیت ۱٦,۰ گیگابایت و سیستم‌عامل (Windows 11 Home 64-bit) پردازنده (x64) است.

جدول (۱): هایپر پارامترها برای الگوریتم GSA

پارامتر	مقدار
اندازه جمعیت α	۲۰
تعداد تولید ذره β	۲۰
ثابت شتاب a_1	۱,۱۰
ثابت شتاب a_2	۱,۱۰
وزن معیار w	۰,۸۹
ماکزیمم ابعاد مقدار sp-conn	٦۳

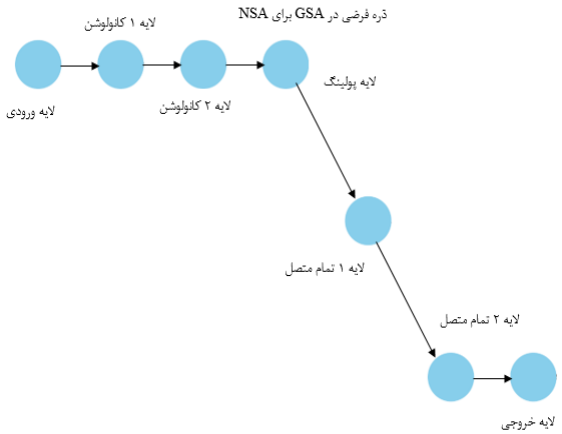
ابزارهای نرم‌افزاری شامل Python 3.9، TensorFlow 2.6.0، NumPy و CUDA Toolkit 11.3 برای شتاب‌دهی GPU بوده‌اند. برای کاهش هزینه محاسبات، روش‌هایی مانند توقف زودهنگام و کاهش ابعاد داده‌ها اعمال‌شده است. این موارد کارایی الگوریتم را در محیط‌های استاندارد به‌طور قابل‌توجهی افزایش می‌دهد. GSANAS با استفاده از کدگذاری طول متغیر و بهینه‌سازی دوسطحی، سرعت همگرایی و کاهش پیچیدگی محاسباتی را تضمین می‌کند. علاوه بر این، استفاده از تکنیک‌های مقیاس‌پذیری معماری و کاهش نمونه‌برداری، امکان جستجوی معماری‌های فشرده و کارآمد را فراهم کرده است.

٤-۲ نتایج پیاده‌سازی

در این قسمت، به ارائه نتایج تجربی بر روی مدل پیشنهادی GSANAS می‌پردازیم. ابتدا، عملکرد مدل پیشنهادی را در مقایسه با روش‌های پیشین بر روی مجموعه داده‌های CIFAR-10 و CIFAR-100 بررسی می‌کنیم. سپس، با استفاده از روش‌های بصری، روند تکامل ساختار شبکه در طول اجرای الگوریتم را به‌صورت گرافیکی نمایش می‌دهیم و به تحلیل همگرایی ذرات می‌پردازیم. همچنین، برای ارزیابی قابلیت اطمینان مدل، به بررسی همبستگی بین نتایج پیش‌بینی‌شده و داده‌های واقعی می‌پردازیم. درنهایت، برای بررسی قابلیت تعمیم‌پذیری مدل، معماری‌های شبکه‌ای به‌دست‌آمده را بر روی مجموعه داده ImageNet نیز آزمایش می‌کنیم. درروش پیشنهادی، داده‌های تست به‌صورت جداگانه برای ارزیابی نهایی عملکرد معماری‌های منتخب استفاده می‌شوند. این داده‌ها در فرآیند جستجوی معماری یا تنظیم مدل (Training)مورداستفاده قرار نمی‌گیرند، بلکه صرفاً جهت بررسی تعمیم‌پذیری مدل بر روی‌داده‌هایی که در فرآیند آموزش دخیل نبوده‌اند، به کار می‌روند. بدین ترتیب ارزیابی عملکرد مدل، بدون تأثیر از داده‌های دیده‌شده در مراحل قبلی صورت می‌گیرد.

جدول ۲ مقایسه بین عملکرد مدل پیشنهادی GSANAS و سایر روش‌های موجود بر روی مجموعه داده‌های CIFAR-10 و CIFAR-100 ارائه می‌دهد. این مقایسه بر اساس معیارهایی مانند تعداد پارامترها و زمان جستجو برحسب روزهای GPU انجام‌شده است. نتایج نشان می‌دهد که مدل‌های GSANAS-A، GSANAS-B و GSANAS-C به ترتیب از کوچک به بزرگ، دارای کمترین تعداد پارامتر در مقایسه با سایر روش‌ها هستند. این نشان‌دهنده کارایی بالای روش پیشنهادی دریافتن معماری‌های فشرده و درعین‌حال قدرتمند است. الگوریتم جستجوی GSA دوسطحی پیشنهادی به‌طور قابل‌توجهی سریع‌تر از سایر روش‌های جستجوی معماری عمل می‌کند.

شکل (٦) یک‌ذره فرضی در الگوریتم جستجوی گرانشی (GSA) برای طراحی معماری شبکه‌های عصبی (NAS) را نمایش می‌دهد. هر گره نمایانگر یک‌لایه شبکه عصبی مانند لایه ورودی، کانولوشنی، تجمعی یا کاملاً متصل است و یال‌ها ارتباط بین لایه‌ها را نشان می‌دهند. این ساختار بیانگر نحوه تعریف و بهینه‌سازی معماری شبکه در فرآیند جستجوی الگوریتم GSA است.



شکل (۶): نمایش یک‌ذره فرضی در الگوریتم جستجوی گرانشی برای طراحی معماری عصبی (GSA-NAS).

علاوه بر سرعت‌بالا، معماری‌های به‌دست‌آمده توسط این الگوریتم نیز ازنظر عملکرد بسیار خوب است، به‌ویژه GSANAS-C که باوجود داشتن تعداد پارامترهای کم، نتایج بسیار خوبی را ارائه می‌دهد. برای مقایسه، پنج معماری CNN را انتخاب کردیم که با استفاده از تکنیک‌های فشرده‌سازی طراحی‌شده‌اند. اگرچه این معماری‌ها تلاش کرده‌اند تا به مدل‌های فشرده دست یابند، اما نتایج نشان می‌دهد که مدل‌های پیشنهادی GSANAS-A، GSANAS-B و GSANAS-C به‌طور قابل‌توجهی کارآمدتر هستند. مدل‌های پیشنهادی نه‌تنها تعداد پارامترهای کمتری دارند، بلکه دقت طبقه‌بندی آن‌ها نیز بالاتر است. جستجوی خودکار معماری (NAS)روشی کارآمد برای یافتن بهترین ساختار شبکه است.

روش پیشنهادی GSANAS-C، با بهره‌گیری از جستجوی خودکار، موفق شده است مدل‌هایی با دقت بالا و اندازه کوچک‌تر نسبت به روش‌های سنتی ایجاد کند. این روش نه‌تنها در مجموعه داده‌های استاندارد عملکرد خوبی نشان داده، بلکه هزینه محاسباتی آن نیز بسیار کمتر است. همچنین، GSANAS-C به دخالت انسانی کمتری نیاز دارد و به‌طور خودکار می‌تواند معماری‌های بهینه را کشف کند. به‌منظور بررسی عملکرد پایداری روش پیشنهادی، میانگین نتایج ده اجرای مستقل بر روی هر دو مجموعه داده در جدول ۳ گزارش‌شده است. همان‌طور که مشاهده می‌شود، دقت متوسط سه معماری مختلف در مقیاس‌های مختلف بسیار قابل‌قبول بوده و با بهترین روش‌های موجود قابل‌مقایسه است.

بااین‌حال، در مجموعه داده CIFAR-100 به دلیل پیچیدگی محاسباتی، نوسانات بیشتری در دقت دارد.

جدول (۲): مقایسه عملکرد GSANAS با روش‌های پیشرفته ازنظر دقت طبقه‌بندی (٪)، تعداد پارامترها و هزینه جستجو در مجموعه داده‌های CIFAR-10 و CIFAR-100

روش	تعداد پارامترها	CIFAR-10	CIFAR-100	هزینه جستجو
resnet-110 [۲۵]	۱٫۷M	۹۳٫۵۷	۷٤٫۸٤	–
MobileNetV2 [۱۹]	۲٫۱M	۹٤٫۵٦	۷۷٫۰۹	–
NASNet-A [۲۶]	۳٫۳M	۹۷٫۳۵	–	۲۰۰۰
FPSO [۲۷]	۰٫۷M	۹۳٫۷۲	–	۱٫٦۵
CNN-GA [۲۸]	۲٫۹M	۹٦٫۷۸	–	۳۵
GSANAS-A	۰٫۱۲M	۹۷٫۰۱	۸۲٫۲۰	۱٫۰۹
GSANAS-B	۰٫۳۱M	۹۷٫۲۱	۸۳٫٤۳	۱٫۰۹
GSANAS-C	۱٫٦M	۹۸٫٤۸	۸۵٫۱۲	**۱٫۰۵**

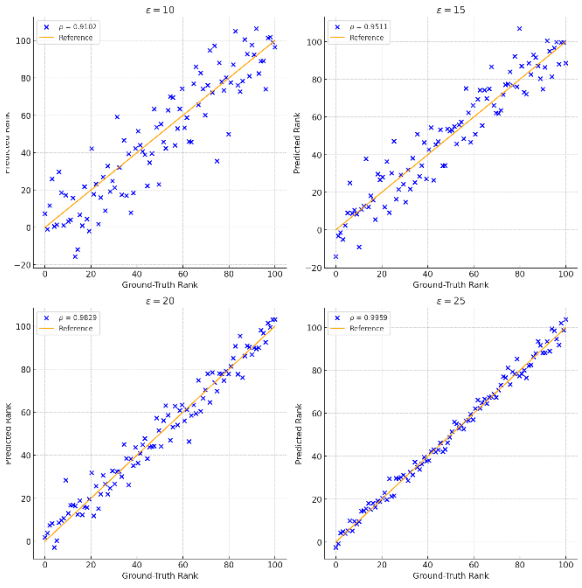
ارزیابی پارامترهای معماری‌های به‌دست‌آمده در هر دو مجموعه داده نشان می‌دهد که الگوریتم GSANAS به‌طور کارآمد توانسته است معماری‌هایی با تعداد پارامترهای تقریباً یکسان و بسیار بهینه را کشف کند.

جدول (۳): خلاصه دقت مدل GSANAS روی دو مجموعه داده

مجموعه داده	تعداد پارامترها	مقیاس	دقت
CIFAR-10	۰٫۱۱M ۰٫۳٦M ۰٫۳۲M	A B C	۰٫۰۱+۹٦٫۱۱ ۰٫۰۲+۹۸٫۲۵ ۰٫۰۱+۹۸٫٦۹
CIFAR-100	۰٫۱۱M ۰٫۳۷M ۱٫۳۲M	A B C	۰٫۰۱+۸۰٫۱۱ ۰٫۰۲+۸۱٫۲۵ ۰٫۰۱+۸۲٫٦۹

این نشان‌دهنده توانایی بالای این الگوریتم در جستجوی معماری‌های CNN فشرده است. همان‌طور که در شکل ٦ نشان

داده‌شده است، ρزمانی به بالاترین مقدار خود می‌رسد که $\varepsilon = 20$ باشد و الگوریتم پیشنهادی نیز قابل‌اعتمادترین الگوریتم در پیش‌بینی عملکرد معماری‌های شبکه است که قابلیت اعتماد شتاب‌دهی یکپارچه در GSANAS را نشان می‌دهد. روش پیشنهادی GSANAS بر روی مجموعه داده‌های CIFAR-10، CIFAR-100 و ImageNet مورد آزمایش قرار گرفت. نتایج نشان داد که این روش با دقت طبقه‌بندی ۹۸,۴۸٪ در CIFAR-10، ۸۵,۱۲٪ در CIFAR-100 و عملکرد برجسته در ImageNet، ضمن کاهش قابل‌توجه هزینه‌های محاسباتی 1.05 روز GPU برای جستجو و تعداد پارامترها، برتری چشمگیری نسبت به روش‌های پیشرفته دیگر دارد. این مدل همچنین توانایی بالایی در طراحی معماری‌های فشرده با حداقل پیچیدگی محاسباتی ارائه می‌دهد. این نتایج برتری روش GSANAS را ازنظر دقت طبقه‌بندی، کاهش تعداد پارامترها و هزینه جستجوی معماری بر روی داده‌های استاندارد نشان می‌دهد.



شکل (٦): همبستگی رتبه ρ با تعداد مختلف دوره‌های آموزشی توقف زودهنگام ε.

نتایج معماری پیشنهادی در مقایسه با معماری‌های پیشرفته نشان می‌دهد که روش GSA-NAS به دلیل طراحی هوشمندانه فضای جستجو و بهینه‌سازی ساختار شبکه، دقت طبقه‌بندی بهتری را ارائه می‌دهد. علاوه بر این، هزینه جستجوی معماری در مقایسه با الگوریتم‌های سنتی کاهش قابل‌توجهی داشته است، به‌گونه‌ای که در مقایسه با NASNet، هزینه جستجو حدود ۵۰ درصد کمتر و دقت نهایی به‌طور میانگین ۱,۲ درصد بیشتر است.

۵- نتیجه‌گیری

این مقاله، الگوریتمی به نام GSANAS را معرفی می‌کند که با استفاده از روش جستجوی گرانشی (GSA)، معماری‌های شبکه‌های عصبی کانولوشنی (CNN) فشرده‌ای را برای طبقه‌بندی تصاویر طراحی می‌کند. GSANAS با استفاده از بلوک‌های اصلاح‌شده MBConv، فضای جستجویی کارآمد برای کشف معماری‌های بهینه ایجاد می‌کند. همچنین، از یک الگوریتم GSA دوسطحی انعطاف‌پذیر برای بهینه‌سازی هم‌زمان پیکربندی‌های شبکه و اتصالات بین گره‌ها استفاده می‌شود. این الگوریتم با ترکیب هوشمندانه روش‌های توقف زودهنگام، کاهش نمونه‌برداری و ساده‌سازی معماری، پیچیدگی محاسباتی را به‌طور قابل‌توجهی کاهش می‌دهد. GSANAS قادر است به‌صورت خودکار و هم‌زمان، عمق، عرض، وضوح فضایی و نحوه اتصال لایه‌های شبکه را جستجو کند. نتایج آزمایش‌ها بر روی مجموعه داده‌های استاندارد نشان می‌دهد که GSANAS در مقایسه با سایر روش‌های طراحی معماری، عملکرد بسیار بهتری داشته و به پارامترهای محاسباتی کمتری نیاز دارد. بااین‌حال، این الگوریتم قادر به کنترل دقیق تعداد پارامترهای شبکه نیست که ممکن است در کاربردهای با محدودیت سخت‌افزاری مشکل‌ساز باشد. جهت بهبود این الگوریتم، می‌توان آن را برای طراحی انواع دیگر شبکه‌های عصبی فشرده مانند شبکه‌های بازگشتی (RNN) و شبکه‌های مبتنی بر توجه به کار برد. همچنین، می‌توان مسئله طراحی معماری فشرده را به‌عنوان یک مسئله بهینه‌سازی چندهدفه فرموله کرد تا تعادل بهتری بین اهداف مختلف حاصل شود. یکی دیگر از جهت‌های تحقیقاتی آینده، بهبود روش جستجوی معماری در مجموعه داده‌های بزرگ مانند ImageNet است.

References

[1] L. Alzubaidi et al., “Review of deep learning: concepts, CNN architectures, challenges, applications, future directions,” J. Big Data, vol.

[13] B. Koonce and B. Koonce, “MobileNetV3,” in Convolutional Neural Networks with Swift for Tensorflow, Berkeley, CA: Apress, 2021, pp. 125-144.
[14] Z. Lu et al., “NSGA-Net: neural architecture search using multi-objective genetic algorithm,” 2019, pp. 419-427.
[15] Z. Yue, B. Lin, Y. Zhang, and C. Liang, “Effective, Efficient and Robust Neural Architecture Search,” 2022, pp. 1-8.
[16] S. Li, Y. Sun, G. G. Yen, and M. Zhang, “Automatic Design of Convolutional Neural Network Architectures Under Resource Constraints,” IEEE Trans. Neural Netw. Learn. Syst., pp. 1-15, 2021.
[17] O. Avalos, “GSA for machine learning problems: A comprehensive overview,” Appl. Math. Model., vol. 92, pp. 261-280, Apr. 2021.
[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” Commun. ACM, vol. 60, no. 6, pp. 84–90, May 2017, doi: 10.1145/3065386.
[19] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, “MobileNetV2: Inverted Residuals and Linear Bottlenecks,” in 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT: IEEE, Jun. 2018, pp. 4510–4520. doi: 10.1109/CVPR.2018.00474.
[20] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, “Densely Connected Convolutional Networks,” in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI: IEEE, Jul. 2017, pp. 2261–2269. doi: 10.1109/CVPR.2017.243.
[21] [29] A. Krizhevsky, G. Hinton et al., “Learning multiple layers of features from tiny images,” University of Toronto, 2009.
[22] O. Russakovsky et al., “ImageNet Large Scale Visual Recognition Challenge,” 2014, arXiv. doi: 10.48550/ARXIV.1409.0575.
[23] H. Mittal, A. Tripathi, A. C. Pandey, and R. Pal, “Gravitational search algorithm: a comprehensive analysis of recent variants,” Multimed Tools Appl, vol. 80, no. 5, pp. 7581–7608, Feb. 2021, doi: 10.1007/s11042-020-09831-4.
[24] Z. Brahmi and R. Derouiche, “EQGSA-DPW: A Quantum-GSA Algorithm-Based Data Placement for Scientific Workflow in Cloud Computing Environment,” J Grid Computing, vol. 22, no. 3, p. 57, Sep. 2024, doi: 10.1007/s10723-024-09771-5.
[25] K. He, X. Zhang, S. Ren, and J. Sun, “Deep
8, no. 1, Mar. 2021.
[2] H.-C. Shin et al., “Deep Convolutional Neural Networks for Computer-Aided Detection: CNN Architectures, Dataset Characteristics and Transfer Learning,” IEEE Trans. Med. Imaging, vol. 35, no. 5, pp. 1285-1298, May 2016.
[3] M. Hussain, J. J. Bird, and D. R. Faria, “A Study on CNN Transfer Learning for Image Classification,” in Advances in Computational Intelligence Systems, vol. 840, A. Lotfi, H. Bouchachia, A. Gegov, C. Langensiepen, and M. McGinnity, Eds. Cham: Springer International Publishing, 2019, pp. 191-202.
[4] P. Ren et al., “A Comprehensive Survey of Neural Architecture Search: Challenges and Solutions,” ACM Comput. Surv., vol. 54, no. 4, pp. 1-34, May 2022.
[5] J. Huang, B. Xue, Y. Sun, M. Zhang, and G. G. Yen, “Particle Swarm Optimization for Compact Neural Architecture Search for Image Classification,” IEEE Trans. Evol. Comput., pp. 1-1, 2022.
[6] E. Rashedi, H. Nezamabadi-pour, and S. Saryazdi, “GSA: A Gravitational Search Algorithm,” Inf. Sci., vol. 179, no. 13, pp. 2232-2248, Jun. 2009.
[7] S. Tabatabaei, “A new gravitational search optimization algorithm to solve single and multiobjective optimization problems,” J. Intell. Fuzzy Syst., vol. 26, no. 2, pp. 993-1006, 2014.
[8] D. Pelusi, R. Mascella, and L. Tallini, “Revised Gravitational Search Algorithms Based on Evolutionary-Fuzzy Systems,” Algorithms, vol. 10, no. 2, p. 44, Apr. 2017.
[9] E. Rashedi, E. Rashedi, and H. Nezamabadi-pour, “A comprehensive survey on gravitational search algorithm,” Swarm Evol. Comput., vol. 41, pp. 141-158, Aug. 2018.
[10] “Spacetime and Geometry: An Introduction to General Relativity,” ResearchGate. [Online]. Available: https://www.researchgate.net/publication/230967918_Spacetime_and_Geometry_An_Introduction_to_General_Relativity. [Accessed: 24-Sep-2021].
[11] O. Russakovsky et al., “ImageNet Large Scale Visual Recognition Challenge,” Int. J. Comput. Vis., vol. 115, no. 3, pp. 211-252, Dec. 2015.
[12] Y. Wang, J. Yan, Q. Sun, J. Li, and Z. Yang, “A MobileNets Convolutional Neural Network for GIS Partial Discharge Pattern Recognition in the Ubiquitous Power Internet of Things Context: Optimization, Comparison, and Application,” IEEE Access, vol. 7, pp. 150226-150236, 2019.

Residual Learning for Image Recognition," in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA: IEEE, Jun. 2016, pp. 770–778. doi: 10.1109/CVPR.2016.90.

[26] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning Transferable Architectures for Scalable Image Recognition," in 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT: IEEE, Jun. 2018, pp. 8697–8710. doi: 10.1109/CVPR.2018.00907.

[27] J. Huang, B. Xue, Y. Sun, and M. Zhang, "A Flexible Variable-length Particle Swarm Optimization Approach to Convolutional Neural Network Architecture Design," in 2021 IEEE Congress on Evolutionary Computation (CEC), Kraków, Poland: IEEE, Jun. 2021, pp. 934–941. doi: 10.1109/CEC45853.2021.9504716.

[28] Y. Sun, B. Xue, M. Zhang, and G. G. Yen, "Completely Automated CNN Architecture Design Based on Blocks," IEEE Trans. Neural Netw. Learning Syst., vol. 31, no. 4, pp. 1242–1254, Apr. 2020, doi: 10.1109/TNNLS.2019.2919608.

Journal of Applied And Basic Machine Intelligence Research (ABMIR)

Applied and Basic Machine Intelligence Research

Vol. 2, No. 1, Spring & Summer 2024, pp. 77 - 91

Compact Neural Architecture Search for Image Classification Using Gravitational Search Algorithm

Sajad Bastami , Mohammad Bagher Dowlatshahi*

Computer Engineering Department, Lorestan University, Khoramabad, Iran

Article Information

Original Research Paper

Received:
2024 October 4

Accepted:
2024 December 31

Keywords:
Convolutional Neural Networks (CNNs), Deep Learning, Gravitational Search Algorithm (GSA), Neural Architecture Search (NAS)

Corresponding Author*:
dowlatshahi.mb@lu.ac.ir

Abstract

This paper presents a compact neural architecture search method for image classification using the Gravitational Search Algorithm (GSA). Deep learning, through multi-layer computational models, enables automatic feature extraction from raw data at various levels of abstraction, playing a key role in complex tasks such as image classification. Neural Architecture Search (NAS), which automatically discovers new architectures for Convolutional Neural Networks (CNNs), faces challenges such as high computational complexity and costs. To address these issues, a GSA-based approach has been developed, employing a bi-level variable-length optimization technique to design both micro and macro architectures of CNNs. This approach, leveraging a compact search space and modified convolutional bottlenecks, demonstrates superior performance compared to state-of-the-art methods. Experimental results on CIFAR-10, CIFAR-100, and ImageNet datasets reveal that the proposed method achieves a classification accuracy of 98.48% with a search cost of 1.05 GPU days, outperforming existing algorithms in terms of accuracy, search efficiency, and architectural complexity.

: 10.22034/abmir.2024.22228.1066

E-ISSN: 2821-2037