تاریخ دریافت مقاله: ۹۸/۰۱/۱۸
تاریخ پذیرش مقاله: ۹۸/۰۶/۱۲

به‌کارگیری پردازنده گرافیکی در یادگیری ماشین: تأکید بر شبکه‌های عصبی و یادگیری عمیق

فاطمه ناظمی جنابی*
مرکز رشد واحدهای فناور ـ دانشگاه بین المللی امام خمینی(ره) ـ قزوین ـ ایران
پست الکترونیکی: nazemijenabi@org.ikiu.ac.ir

حمیدرضا حمیدی
دانشکده فنی و مهندسی ـ دانشگاه بین المللی امام خمینی(ره) ـ قزوین ـ ایران
پست الکترونیکی: hamidreza.hamidi@eng.ikiu.ac.ir

چکیده:

این مقاله علاوه بر مرور ساختار پردازشی و قابلیت‌های پردازنده گرافیکی به تسریع‌های به‌دست آمده برای محاسبات عمومی و به‌طور خاص، شبکه‌های عصبی و یادگیری عمیق می‌پردازد. توسعه ابزارها، کتابخانه‌ها و چارچوب‌های یادگیری ماشین برای پردازنده گرافیکی همچنان نیازمند توسعه است. برای مقیاس‌پذیری روی چند پردازنده گرافیکی، موضوعاتی نظیر پشتیبانی کتابخانه‌ها و ابزارها از ارتباطات میان پردازنده‌های گرافیکی و رفع نیاز به بازنویسی کد توسط کاربر نیازمند توجه هستند. یکی از موانع محاسبات حجیم از جمله یادگیری عمیق، انرژی‌بر بودن آن‌هاست و گرایش فعلی به سمت طراحی کمک‌پردازنده‌های[1] اختصاصی است. سازندگان پردازنده‌های گرافیکی نیز با معرفی ریزمعماری وُلتا انویدیا، یک رویکرد سخت‌افزار اختصاصی در هوش مصنوعی و یادگیری ماشین در پیش گرفته‌اند. این‌گونه معماری پردازنده‌های گرافیکی (وُلتا و تورینگ) در مقایسه با سایر شتاب‌دهنده‌های سخت‌افزاری نظیر اف.پی.جی.اِی. و اِی.سیک.[2]، از حیث کارآمدی انرژی و کارایی اجرای الگوریتم‌های یادگیری عمیق قابل رقابت هستند.

واژه‌های کلیدی: پردازنده گرافیکی، یادگیری ماشین، شبکه عصبی مصنوعی، پردازش موازی، کودا

۱- مقدمه

تلاش برای افزایش قدرت پردازشی و کاهش رکود دستیابی به حافظه، در ابتدا با بهینه‌سازی پردازنده مرکزی تامین می‌گردید ولی به تدریج با محدودیت مواجه شد [۱]. در مواجهه با محدودیت‌های بهبود کارایی برنامه‌های ترتیبی اجرا شده روی پردازنده‌های تک‌هسته‌ای، تمایل از افزایش کارایی یک هستهٔ پردازنده، به سمت افزایش تعداد هسته‌ها رفت [۲]. هرچند انتظارات از ظرفیت رشد رویکرد چند هسته‌ای نیز به‌نظر رو به افول است و پدیده سیلیکون تاریک[3]، امکان رشد بیشتر پردازنده‌های عمومی

* نویسندهٔ مسئول

1- Coprocessor

2-Field Programmable Gate Arrays (FPGA) & Application Specific Integrated Circuit (ASIC)

3-Dark Silicon

برای شتابدهی به پردازش‌های پرحجم امروزی را با تردید مواجه نموده است [۳].

قدرت پردازشی پردازنده گرافیکی حاصل معماری ویژه آن است که برای دستیابی به حداکثر کارایی روی وظایف بسیار موازیِ گرافیک کامپیوتری توسعه یافته است [۲]. می‌توان گفت، پردازنده گرافیکی در ابتدا با نیاز صنعت بازی‌های رایانه‌ای به پردازش بیدرنگ و کارآمد صحنه‌های سه بعدی پیچیده با دقت بالا و در نرخ قاب‌های تعاملی طراحی شد [٤]. به عبارت دقیق‌تر، پردازنده گرافیکی به‌طور اختصاصی برای پرداز زدن[۴] و سایر برنامه‌های کاربردی گرافیکی که سطح بالایی از موازی بودن داده را دارا هستند، طراحی شد. این پردازنده‌ها، پردازندهٔ توابع ثابت بودند که به تدریج تا حدی تنظیم‌پذیر شده و امروزه در عین قدرتمندی، انعطاف‌پذیر و قابل برنامه‌نویسی نیز هستند [٤].

توانمندی‌های محاسباتی پردازندهٔ گرافیکی، تدریجا به حوزهٔ جی.پی.جی.پی.یو.[۵] یا برنامه‌نویسی عمومی روی پردازنده گرافیکی منتهی شده است. از میان طیف وسیع شاخه‌های علمی که تاکنون از توان محاسباتی پردازنده‌های گرافیکی بهره برده‌اند [۵]، در این مقاله به‌طور خاص به حوزهٔ شبکه‌های عصبی و یادگیری عمیق پرداخته شده است.

بهره‌برداری از توان بالقوهٔ پردازشی پردازندهٔ گرافیکی برای محاسبات عمومی، منوط به شناخت مشخصات بنیادین سخت‌افزاری–نرم‌افزاری این پردازنده و ناتوانی‌های ذاتی برخاسته از این طراحی منحصر به فرد است. همچنین کارکردهای مناسب آن در کنار نقاط ضعف و قوتش در مقایسه با پردازنده‌های سنتی و سایر شتاب‌دهنده‌های سخت‌افزاری[۶] نیز باید مدنظر قرار گیرد.

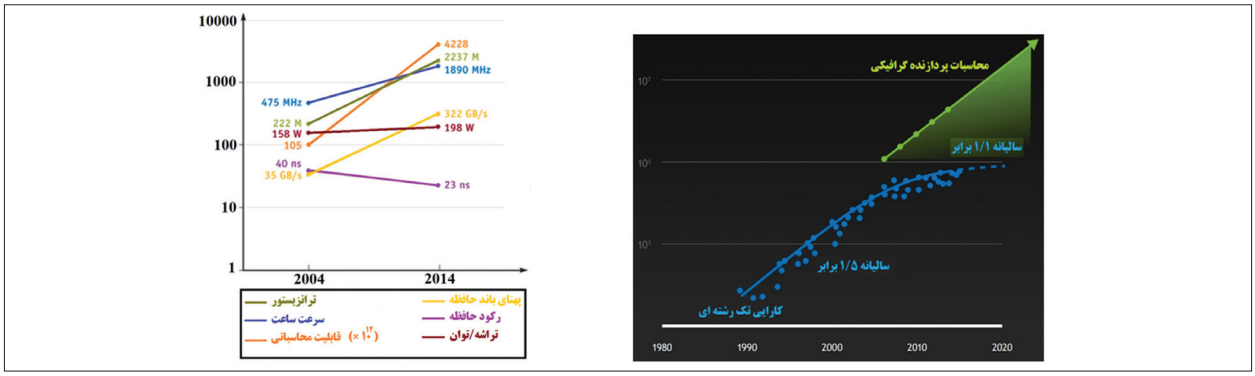
در بخش دوم مقاله، به بررسی روند رشد کارایی پردازنده‌های مرکزی و گرافیکی پرداخته‌ایم. رشد عملیات ممیز شناور و پهنای باند میان پردازنده‌های مرکزی و گرافیکی را مقایسه نموده و خواهیم دید مشخصه‌های کلیدی پردازنده گرافیکی چطور در طول زمان تغییر کرده است. در بخش سوم، پردازش عمومی روی پردازنده گرافیکی مطرح شده و ساختار پردازشی منحصر به فرد این پردازنده مورد بررسی قرار می‌گیرد. در پردازنده گرافیکی، تعداد ترانزیستور بیشتری به پردازش داده تخصیص یافته تا حافظه نهان یا کنترل جریان. بنا به این معماری منحصر به فرد، پردازنده گرافیکی برای محاسبات داده‌موازی مناسب است. هرچند این معماری توان محاسباتی عظیمی را ارائه نموده در عین حال ویژگی‌هایی دارد که موجب می‌شود لزوما همهٔ الگوریتم‌ها نتوانند از این نیروی پردازشی عظیم بهره ببرند. بنابراین شناخت مشخصه‌های الگوریتم‌های مناسب پردازنده گرافیکی حائز اهمیت است. این پردازنده‌ها به تدریج از پردازنده‌های توابع ثابت به پردازنده‌های کاملا قابل برنامه‌نویسی برای محاسبات عمومی تبدیل شدند به‌طوری که امروزه روند رشد تعداد برنامه‌های کاربردی پیاده‌سازی شده روی پردازنده گرافیکی و تسریع‌های حاصله بسیار چشمگیر است. با توسعه کاربردهای عمومی این پردازنده‌ها، واسطهای برنامه‌نویسی جدیدی نظیر کودا نیز ارائه شده و موضوعاتی همچون کتابخانه‌ها، ابزارهای اشکال‌زدایی و پروفایل‌سازی نیز بیشتر مورد توجه قرار گرفته است.

در بخش چهارم به پیاده‌سازی شبکه‌های عصبی مصنوعی روی پردازنده‌های گرافیکی می‌پردازیم. پیچیدگی محاسباتی شبکه عصبی بازگشتی، آموزش کارآمد این شبکه‌ها را دشوار ساخته بود ولی با ظهور پردازنده‌های گرافیکی، آموزش کارآمد شبکه‌های بزرگ عصبی بازگشتی میسر شد. امروزه تقریبا تمام گروه‌هایی که روی آموزش شبکه‌های عمیق کار می‌کنند، از پردازنده گرافیکی استفاده می‌کنند. گستردگی استفاده از پردازنده گرافیکی به اندازه‌ای است که پشتیبانی از پردازنده گرافیکی در جعبه ابزار پردازش موازی متلب در نظر گرفته شده است. در

۴- Rendering: عبارتست از فرآیند ساخت یک تصویر از روی یک مدل به‌وسیلهٔ برنامه‌های کامپیوتری.

5- General-Purpose computation on Graphics Processing Units (GPGPU)

6- Accelerator



شکل ۱: مقایسه رشد کارایی پردازنده مرکزی و گرافیکی [۱۲] (راست) و تغییرات مشخصه‌های کلیدی پردازنده گرافیکی [۱۱] (چپ)

این بخش، به معرفی جعبه ابزارها و سایر امکانات موجود در متلب برای استفاده از توان پردازش موازی پردازنده مرکزی برای حوزه یادگیری ماشین می‌پردازیم. همچنین، چارچوب‌های رایج یادگیری عمیق را بررسی و مقایسه می‌کنیم. سپس پیشنهادهایی را برای بهره‌گیری بهتر از ظرفیت‌های فعلی مطرح نموده و نکاتی را که توجه به آن‌ها برای پاسخگویی به نیاز روزافزون به توان محاسباتی بالاتر برای حجم‌های عظیم‌تر داده در آینده می‌تواند مفید باشد مطرح می‌کنیم. این نکات عبارتند از توسعه ابزارها، کتابخانه‌ها و چارچوب‌های الگوریتم‌های یادگیری ماشین برای پردازنده گرافیکی، به اشتراک‌گذاری پیاده‌سازی‌ها به‌صورت متن‌باز و مصرف انرژی.

۲-روند رشد پردازنده‌ها

رشد کارایی کامپیوترها تنها بین سال‌های ۱۹۸۵-۱۹۹۶ بیش از ۵۰٪ بوده است. این نرخ بین سال‌های ۱۹۹۶-۲۰۰۴ به ۴۱٪ کاهش یافت [۶]. نرخ رشد پردازنده‌های گرافیکی از قانون مور سبقت گرفته است [۷]. این رشد، طی بازه ۴۰ ساله از ۱۹۸۰ تا ۲۰۲۰ در شکل(۱)-راست نشان داده شده است. پردازنده‌های مرکزی، تعداد هسته‌های بسیار محدودی دارند در حالی‌که پردازنده‌های گرافیکی، هسته‌های بسیار زیادی دارند. رقابت بر سر قدرت محاسبه، علاوه بر محدودیت گنجاندن هسته‌های بیشتر، یک مانع پراهمیت دیگر نیز دارد: مصرف انرژی. شتاب‌دهنده‌هایی همچون کارت‌های گرافیکی، یکی از راه‌کارهای بهینه‌سازی است [۹]. در حالی‌که پردازنده مرکزی به سقف کارایی ترتیبی برخورد کرده است [۱۰]، پردازنده گرافیکی رشد قابل توجهی را در زمینهٔ پهنای باند حافظه و ترانزیستور داشته و در عین حال مصرف انرژی الکتریکی نسبتا ثابت داشته است [۱۱]. این رشد، در شکل (۱)-چپ مشاهده می‌شود.

پهنای باند (خارج از تراشه) حافظه نیز در ابتدا هر دو سال یکبار دو برابر می‌شد ولی این رویه در چند سال اخیر کند شده است. کارایی پردازندهٔ گرافیکی همچنان به رشد ۴۵ درصدی سالیانه برای دقت ساده و نرخی بالاتر برای دقت مضاعف[۷] ادامه می‌دهد [۸]. شکل‌های (۲) و (۳) نمایانگر این موضوع می‌باشند.

۳- پردازش عمومی بر روی پردازنده گرافیکی

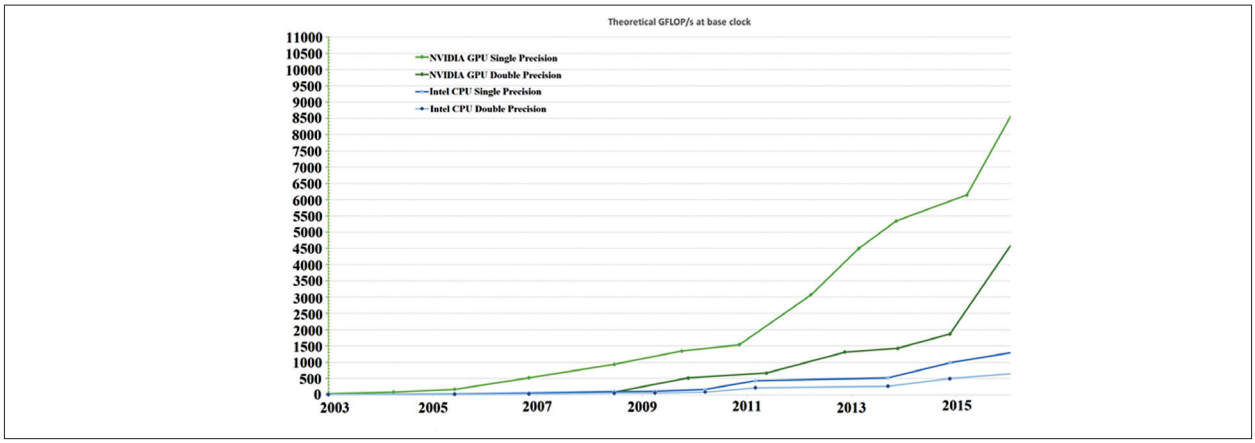
۳-۱- شیوه پردازشی پردازنده گرافیکی

میان ظرفیت ممیزشناور پردازنده‌های مرکزی و گرافیکی تفاوت قابل توجهی وجود دارد. این تفاوت ناشی از طراحی منحصر به فرد پردازنده گرافیکی است که در آن، تعداد ترانزیستور بیشتری به پردازش داده تخصیص یافته تا حافظه نهان یا کنترل جریان [۱۳].

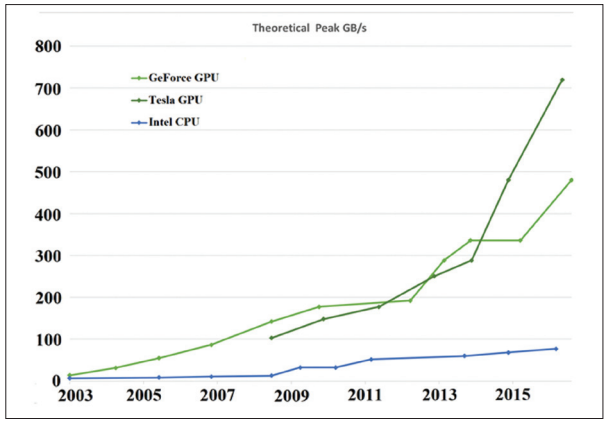
واحدهای قابل برنامه‌سازی پردازنده گرافیکی، مدل برنامه‌سازی اِس.پی.اِم.دی.[۸] را دنبال می‌کنند. این برنامه‌ها

7- Single precision & Double precision

۸- Single Program Multiple Data (SPMD) : روشی که در آن یک برنامه واحد، روی تمام گره‌های یک سیستم موازی بارگذاری شده و هر نسخه از برنامه به‌طور مستقل اجرا می‌شود. بدین ترتیب، جریان دستورالعمل روی هر گره می‌تواند متفاوت باشد.

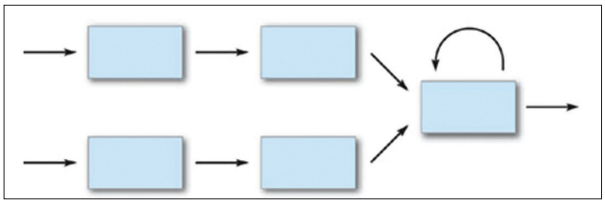


شکل۲: مقایسه رشد سرعت عملیات ممیز شناور پردازنده‌های مرکزی و گرافیکی [۱۳]



شکل۳: مقایسه رشد پهنای باند پردازنده‌های مرکزی و گرافیکی [۱۳]

بدین ترتیب ساختاربندی می‌شوند که تعداد زیادی عنصر داده‌موازی توسط یک برنامهٔ مشترک پردازش می‌شوند [۱٤]. پس پردازنده گرافیکی برای محاسبات داده‌موازی یعنی اجرای موازیِ یک برنامهٔ واحد با حجم محاسباتی بالا بر روی تعداد زیادی عنصر داده‌ای، بسیار مناسب است [۱۵]. به دلیل اجرای یک برنامهٔ واحد روی تک تک عناصر داده‌ای، نیاز چندانی به کنترل جریان پیچیده وجود ندارد. کلید استفاده از پردازنده گرافیکی برای اهدافی غیر از پرداز زدنِ بیدرنگ، در نظر گرفتن آن به عنوان یک محاسبه‌گرِ داده‌موازیِ جریانی است. در مدل جریانی، برنامه‌ها به صورت یک سلسله از عملیات روی جریان‌های داده بیان می‌شوند [۱٦]. این مدل در شکل (٤) نشان داده شده است. مقایسه طراحی و کارکرد پردازنده‌های مرکزی و گرافیکی نیز در جدول (۱) نشان داده شده است.



شکل ۴: برنامه جریانی در قالب گراف وابستگی داده [۱۶]

۳-۲ زبان برنامه‌نویسی و محیط نرم‌افزاری

مدل‌ها و واسط‌های برنامه‌نویسی زیادی طی دهه‌های گذشته برای پردازش عمومی موازی ارائه شده‌اند. رایج‌ترین این واسط‌ها و مدل‌ها عبارتند از اِم.پی.آی.[۹] برای محاسبات خوشه‌ای گسترش‌پذیر، و اُپن.اِم.پی.[۱۰] برای سیستم‌های چند پردازنده‌ای دارای حافظه اشتراکی. در عوض، کودا به عنوان نمونه‌ای از مدل برنامه‌نویسی پردازنده‌های گرافیکی، یک بستر[۱۱] محاسبات موازی است که هم حافظهٔ اشتراکی را تا حدی فراهم می‌کند و هم گسترش‌پذیری عالی دارد [۲].

پردازنده‌های گرافیکی در ابتدا قابلیت برنامه‌نویسی بسیار اندکی داشتند [۹]. با آغاز به‌کارگیری پردازنده‌های گرافیکی برای برنامه‌های کاربردی غیر گرافیکی، لازم بود برنامه کاربردی در قالب عملیات گرافیکی با اُپن.جی.اِل.[۱۲] یا دایرکت‌اِکس[۱۳] بازنویسی شود. سطح انتزاع، به تدریج

۹- Message Passing Interface (MPI): در این مدل، گره‌های یک خوشه دارای حافظهٔ اشتراکی نبوده و تمام تعاملات داده از طریق پیام صورت می‌گیرد.

۱۰- OpenMP: در گسترش‌پذیری محدودیت دارد.

11- Platform

۱۲- OpenGL: توصیفی از یک واسط برنامه‌نویسی برای پرداز زدن گرافیکی.

۱۳- DirectX : مجموعه‌ای از واسط‌های برنامه‌نویسی برای توسعه بازی‌های کامپیوتری و برنامه‌های کاربردی چندرسانه‌ای روی بن‌سازه‌های مایکروسافت.

جدول ۱: مقایسه طراحی پردازنده‌های مرکزی و گرافیکی

پردازنده گرافیکی	پردازنده مرکزی
تعداد زیاد ترانزیستورهای محاسباتی	تخصیص ترانزیستور به وظایف غیر محاسباتی همچون حافظه نهان
دارای هسته‌های زیاد (چند هزار هسته در نسل‌های اخیر)	چند هسته‌ای
محدودیت پهنای باند حافظه و دستیابی سریع به حافظه بَرتراشه	برخورداری از انواع حافظه جهت کاهش رکود دستیابی به حافظه
مخفی‌سازی سربار نقل و انتقالات حافظه با محاسبات نیازمند نرخ بالای محاسبه به تعاملات حافظه	کاهش رکود دستیابی به حافظه به کمک حافظه نهان قوی
بهینه‌سازی شده برای حجم محاسباتی بالا با ماهیت موازی عملیات ممیز شناور	بهینه‌سازی شده برای کارایی بالا روی کدهای ترتیبی امکاناتی نظیر حافظه نهان و پیش‌بینی انشعاب
تمرکز بر توان عملیاتی۱ ترجیحِ افزایش پردازش موازی بر کارایی تک رشته	زمان پاسخ سریع برای هر وظیفه واحد کارایی تک رشته‌ای
طراحی شده برای پرداز زدن	طراحی شده برای انجام امور متعدد
عالی برای موازی‌سازی داده	عالی برای موازی‌سازی وظیفه
افت کارایی در مواجهه با واگرایی	امکان انشعاب مناسب

بـــالا رفت تـــا این‌که امروزه واســـط‌های برنامه‌نویســـی، چارچوب‌ها و ابزارهای سطح بالایی برای این منظور ارائه شده‌اند: دایرکت‌کامپیوت[۱۴]، اُپن.سی.اِل.[۱۵] و کودا [۱۰]. این روند در شکل (۵) نشان داده شده است.

کودا در نوامبر ۲۰۰۶ توسط انویدیا به عنوان یک مدل برنامه‌نویســـی و بستر عمومی محاســـبات موازی معرفی شـــد. برای برنامه‌نویســـان کودا (و همچنین اُپن.سی.اِل.)، پردازنده‌های گرافیکی قابل برنامه‌نویســـی با زبانِ ســـی به‌علاوهٔ افزونه‌ها اســـت. پس نیازی بـــه فهم الگوریتم‌ها و اصطلاحات گرافیکی نیســـت. مدل برنامه‌نویســـی موازی کودا طوری طراحی شـــده تا در عین بســـط موازی‌سازی با افزایش تعداد هســـته‌ها، بار آن را بر دوش برنامه‌نویس نگذارد. یک برنامهٔ ترجمه شـــدهٔ کودا قابل اجرا شدن روی هر تعداد از هســـته است و تنها سیستم در زمان اجرا باید تعـــداد پردازنده‌های فیزیکی را بداند. مدل برنامه‌نویســـی کودا، ناهمگن اســـت. اما یک برنامهٔ کودا، یک کد منبع دارد که هر دو کد میزبان و دستگاه[۱۶] را در بر می‌گیرد. مترجم کـــودا، این دو را حین فرآیند ترجمه جدا می‌کند. بخش‌های ترتیبی از برنامه، روی میزبان اجرا شده و پردازش هسته/کرنل بر عهدهٔ پردازندهٔ گرافیکی اســـت. همچنین فرض بر این است که دستگاه و میزبان هر دو دارای فضای حافظه‌ی جداگانه بر روی دی.رَم.[۱۷] هستند [۱۵].

۳-۳ برنامه‌نویسی عمومی روی پردازنده گرافیکی

صنعت محاسبات با کارایی بالا[۱۸]، به‌واسطهٔ محاسبات شتاب یافته متحول شده است. برخی تسریع‌های حاصل از برنامه‌نویسی عمومی روی پردازنده‌های گرافیکی تا سال ۲۰۱۱ در شـــکل (٦) ارائه شده است. برنامه‌های کاربردی توسعه یافته روی پردازنده گرافیکی، اکنون (مارس ۲۰۱۹) بیش از ششـــصد برنامـــه کاربردی در طیف وســـیعی از کاربردها را پوشش می‌دهد [۵]. روند رشد کمی برنامه‌های کاربردی پیاده‌سازی شده روی پردازنده گرافیکی در شکل (۷) نشان داده شده است.

اصلی‌ترین علل این گرایش رو به رشـــد را می‌توان در قدرت پردازشی عظیم، قیمت مناسب، انعطاف پذیری، قابلیت برنامه‌نویسی و از همه مهم‌تر، در دسترس بودن و توسعهٔ روزافـــزون این پردازنده‌ها- از حیـــث معماری و امکانات نرم‌افزاری- یافت [۷]. پشتیبانی نسبی از شاخه‌ای شدن،

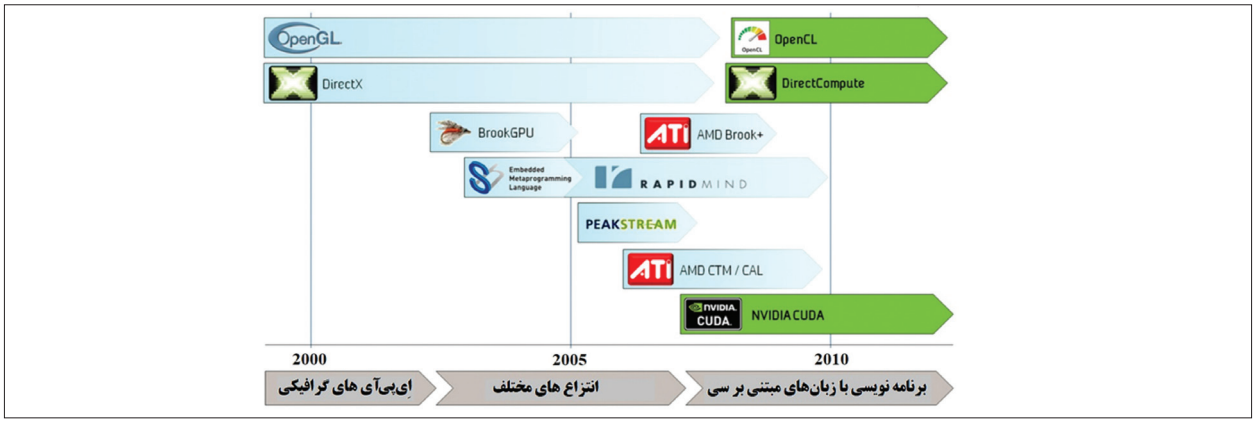
۱۴-DirectCompute: واســـط برنامه‌نویســـی مایکروســـافت بـــرای محاســـبات پردازنده گرافیکی.

۱۵- OpenCL: یک واســـط برنامه‌نویســـی ســـطح پایین برای محاســـبات ناهمگن روی پردازنده‌های گرافیکی.

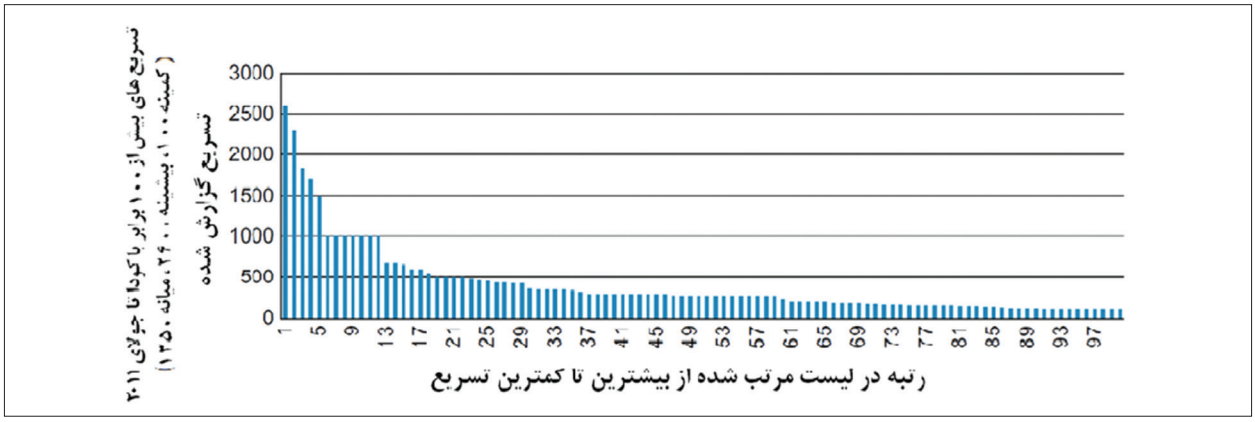
۱۶-Host and Device: منظور از میزبان، پردازنده مرکزی و منظور از دســـتگاه، پردازنده گرافیکی است.

۱۷-Dynamic Random Access Memory (DRAM): مزیتش، ســـادگی ســـاختاری آن است.

۱۸- High Performance computing (HPC): محاســـبات با کارایی بالا به‌طور کلی به حل کارآمد مسائل پیچیده علمی، مهندسی، تجاری و غیره می‌پردازد.



شکل ۵: تاریخچهٔ برنامه‌نویسی پردازنده گرافیکی [۱۰]



شکل ۶: صد تسریع برتر انویدیا در مقایسه با پردازنده مرکزی تا جولای ۲۰۱۱ [۱۶]

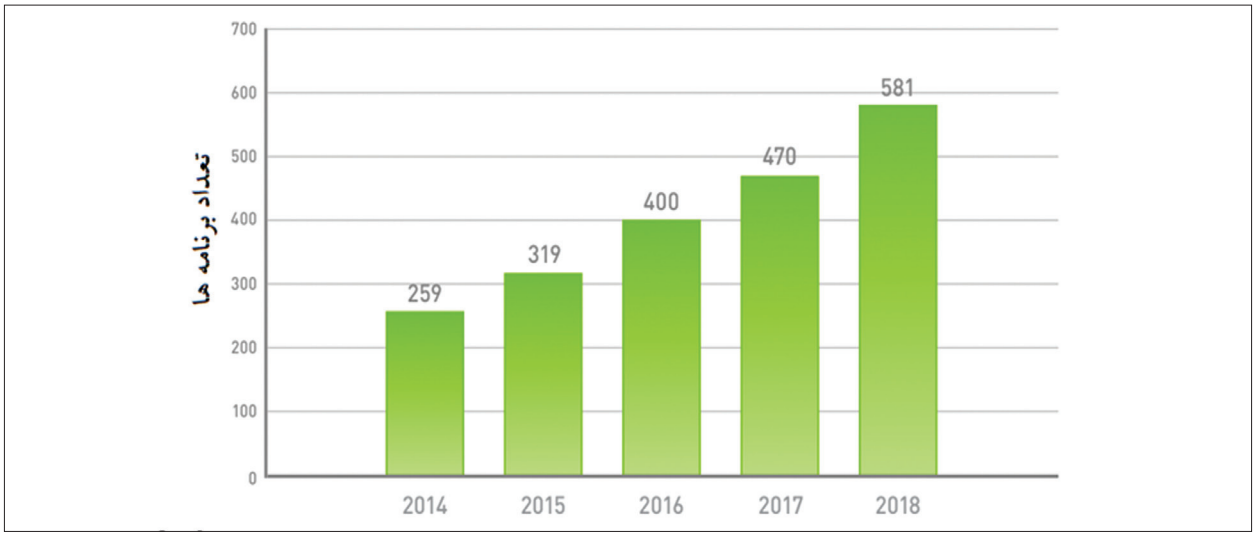
نمونه‌ای از این تغییرات بنیادین است. همچنین، پشتیبانی از استاندارد ممیز شناور[۱۹] و ممیز شناورِ با دقت مضاعف در نسل‌های بعدی پردازنده گرافیکی تامین شده‌اند. رقمی که هسته‌های مدرن پردازنده مرکزی قادر به تحقق‌اش هستند [۲]. از حیث انرژی، کارایی سری کپلر به ازاء هر وات نسبت به سری فرمی سه برابر شده [۱۸]، کارآمدی انرژی سری مَکسوِل نسبت به سری کپلر دو برابر بهتر شد [۱۹] و سخت‌افزار حافظه سه‌بعدی با هدف بهینه‌سازی بیشتر مصرف انرژی در سری پاسکال ارائه شد [۲۰]. پردازنده‌های سری ولتا شرکت انویدیا به‌طور خاص برای الگوریتم‌های هوش مصنوعی طراحی و به بازار عرضه شده است. کارایی یادگیری عمیق این پردازنده‌ها بیش از ۱۲۵ ترافلاپ است که در مقایسه با معماری سری پاسکال، ۵ برابر بهبود یافته است [۲۱].

۱۹- IEEE Floating Point Standard: این استاندارد، دستیابی به نتایج قابل پیش‌بینی در طول پردازنده‌های تولیدکنندگان متفاوت را میسر می‌سازد.

۳-٤- چالش‌های پردازش عمومی روی پردازنده گرافیکی

بخش اعظم دشواری‌های اولیهٔ برنامه‌نویسی عمومی روی پردازنده گرافیکی با توانمندی‌های واسطها و سیستم‌های برنامه‌نویسی جدید نظیر کودا رفع شد اما موضوعاتی همچون اشکال‌زدایی و پروفایل‌سازی همچنان نیازمند توجه و توسعه بیشتر است [۱٤]. امروزه عیب‌یابی و بهینه‌سازی کدهای نوشته شده برای اجرا روی پردازنده‌گرافیکی ساده‌تر شده است. این کار با انویدیا اِن‌سایت[۲۰] و مایکروسافت ویژوال استودیو (برای سیستم‌عامل ویندوز) میسر می‌شود. نسخهٔ اِکلیپسِ بستر توسعهٔ انویدیا اِن‌سایت، یک محیط توسعهٔ مجتمع است که یک محیط مجتمع همه-در-یک را برای ویرایش، ساخت، اشکال‌زدایی و پروفایل‌سازی برنامه‌های کاربردی کودا سی

20- NVIDIA® Nsight™ Eclipse Edition is a full-featured IDE.



شکل ۷: روند رشد تعداد برنامه‌های کاربردی پیاده‌سازی شده روی پردازنده گرافیکی [۱۷]

فراهم می‌کند. انویدیا اِن‌سایت بر ویژوال استودیو منطبق است. این بستر امکان ساخت، اشکال‌زدایی، پروفایل‌سازی و تعقیب برنامه‌های کاربردی ناهمگنِ گرافیکی و محاسباتی را با کودا سی/سی++، اُپن.سی.اِل.، دایرکت کامپیوت، دایرکت تری‌دی[۲۱] و اُپن.جی.اِل. فراهم می‌کند [۲۲]. با وجود ابزارهای موجود، ضرورت بهینه‌سازی کد میان نسل‌های سخت‌افزاری مختلف، بهره‌برداری از توانمندی پردازنده گرافیکی را محدود می‌کند [۲۳].

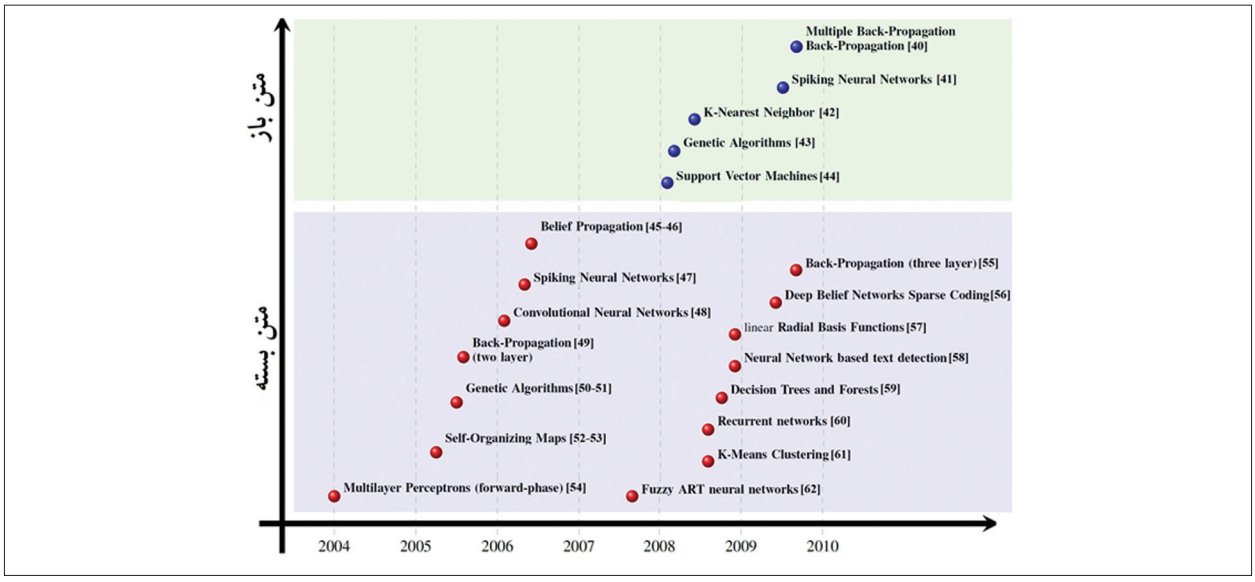
در خصوص سهولت برنامه‌نویسی نیز گرچه کتابخانه‌های رو به توسعهٔ کودا از بار برنامه‌نویسی کاسته‌اند، اما این امکان در پردازنده‌های گرافیکی همچنان ضعیف‌تر از پردازنده‌های مرکزی است [۲۴]. ابزارهای ارائه شده برای پردازنده گرافیکی نیز هنوز نیاز به توجه بیشتری دارد. برخی از ابزارهای ارائه شده عبارتند از: تحلیل خودکار وضعیت رقابت و تداخل بانک حافظهٔ اشتراکی در قالب یک روش تحلیل پویای خودکارِ کودا [۲۵]، یک مدل پیش‌بینی کارایی کودا نیز ارائه شده که با تحلیل شبه‌کد کرنل کودا تخمینی از کارایی را به‌دست می‌دهد [۲۶] و جی.پی.یو.پرف[۲۲] به عنوان چارچوبی برای تعیین قدم‌های بعدی کارایی به‌واسطهٔ شناسایی ریشهٔ اصلی تنگناهای کارایی معرفی شده است [۲۷].

مواردی که تا کنون بدان‌ها اشاره شد مربوط به سخت‌افزار گرافیکی و محیط‌های برنامه‌نویسی و ابزارهای ارائه شده برای آن بود. گذشته از این نکات، الگوی پردازشی پردازنده گرافیکی و سازگاری آن با الگوریتم مورد نظر نیز حائز اهمیت است. با توسعهٔ به‌کارگیری پردازنده‌های گرافیکی در حوزه‌های جدیدتر، برقراری توازن میان معماری ویژهٔ پردازنده گرافیکی و مزایای حاصله‌اش از یک طرف، و برخی محدودیت‌های این معماری برای برنامه‌نویسی عمومی در طرف دیگر، اهمیت بیشتری می‌یابد. امکان شاخه‌ای شدن و افت کارایی ناشی از آن، نمونه‌ای از این وضعیت است [۱۴].

بررسی اهمیت محل نگهداری داده بر کارایی، نشان می‌دهد که نکتهٔ کلیدی در تعیین تسریع قابل دستیابی یک برنامه کاربردی، عبارت است از محل ذخیره‌سازی داده [۲۸]. به‌طوری که بدون اشاره به سربار نقل و انتقالات حافظه، مقایسه میان کارایی پردازنده مرکزی و پردازنده گرافیکی، ناقص است. موضوع محدودیت نقل و انتقال داده در یک سیستم ترکیبی متشکل از چندین پردازنده گرافیکی و مرکزی نیز یکی از موانع جدی کارایی این سیستم‌هاست [۲۹]. بنابراین زمان‌بندی برنامه‌های موازی روی معماری‌های ترکیبی باید به نحوی باشد که توان

۲۱- Direct۳D: یک واسط برنامه‌نویسی سطح پایین که امکان بهره‌برداری از کارت‌های گرافیکی را فراهم می‌کند.

۲۲- GPUPerf: این چارچوب هم تنگناهای موجود در کد را تعیین می‌کند و هم مزایای رفع آن‌ها برای کارایی را ارزیابی می‌نماید.



شکل ۸:پیاده‌سازی الگوریتم‌های یادگیری ماشین روی پردازنده گرافیکی به ترتیب تاریخی [۶۸]

بالای پردازشی پردازنده گرافیکی تحت تأثیر سربار انتقال حافظه قرار نگیرد. به‌علاوه، هنگام برنامه‌نویسی روی کودا تعیین اندازهٔ رشته بلوک‌ها بر کارایی نهایی بسیار مؤثر بوده و نیازمند بررسی است [۳۰].

لزوما تمامی برنامه‌های کاربردی، مناسب برای اجرا روی پردازنده گرافیکی نبوده و چه بسا با افت کارایی مواجه شوند. ماهیت و ساختار برخی الگوریتم‌ها برای پردازنده‌های گرافیکی مناسب نیست. برخی از آن‌ها با تغییر در الگوی محاسبه، قابل انطباق بر پردازنده گرافیکی هستند. مرتب‌سازی، مثالی از الگوریتم قابل انطباق بر پردازنده گرافیکی، و «دنبال کردن اشاره‌گر» نیز مثالی از الگوریتم غیرقابل انطباق بر پردازنده گرافیکی است [۷].

الگوریتم‌های تکاملی به دلیل زمان‌بر بودن و نیاز به منابع محاسباتی بالا، گزینه مناسبی برای پیاده‌سازی روی پردازنده گرافیکی هستند. در بخش بعدی به بهره‌گیری از توان پردازشی پردازنده گرافیکی برای پیاده‌سازی شبکه‌های عصبی مصنوعی می‌پردازیم.

٤- پیاده‌سازی شبکه‌های عصبی مصنوعی روی پردازنده گرافیکی

پس از معرفی کودا در سال ۲۰۰۷، آونز [۷] به مرور پردازش عمومی روی پردازنده گرافیکی پرداخت اما در این مقاله، اشاره‌ای به شبکه‌های عصبی نشد. در همان سال، مقاله دیگری [۳۱] این جای خالی را پوشش داد و به بررسی روش‌ها، چالش‌ها و محدودیت‌های به‌کارگیری پردازنده گرافیکی برای پیاده‌سازی شبکه‌های عصبی پرداخت. این مقاله [۳۱] نتیجه می‌گیرد که پیاده‌سازی الگوریتم‌های شبکه عصبی مصنوعی روی پردازنده گرافیکی، می‌تواند به بهبود چشمگیر کارایی (تا ۲۰۰ برابر) بینجامد ولی چون مربوط به سال ۲۰۰۷ می‌باشد و کاربرد پردازنده گرافیکی در محاسبات عمومی از آن زمان تا کنون دگرگون شده است، مرور مجدد وضع موجود ضروری است. کارهای جدیدتر نیز تقریبا همگی یا ۱) به جنبه خاصی از موضوع می‌پردازند مثلا ابزارهای محاسبات پردازنده گرافیکی برای یادگیری ماشین [۳۲-۳۵]، استفاده از پردازنده گرافیکی برای کلان داده [۳٦]، مقایسه با سایر شتاب‌دهنده‌های سخت‌افزاری [۳۷] یا ۲) پردازنده گرافیکی بخش کوچکی از مبحث اصلی آن‌ها را تشکیل می‌دهد مانند [۳۸, ۳۹]. بنابراین در این مقاله قصد داریم اطلاعات نسبتا جامع و بروزی را از ظرفیت پردازنده گرافیکی برای حوزه یادگیری ماشین و به‌خصوص یادگیری عمیق ارائه کنیم. پیاده‌سازی الگوریتم‌های یادگیری ماشین [٤٠-٦٢] روی

پردازنده گرافیکی تا سال ۲۰۱۰ به ترتیب تاریخی در شکل (۸) نشان داده شده است.

برای دستیابی به کارایی مطلوب‌تر، بهتر است از کارایی هر دو پردازنده مرکزی و گرافیکی و بنا به تناسب ساختار پردازشی هر پردازنده با وظیفه مورد نظر بهره برد. پیاده‌سازی شناسایی متن مبتنی بر شبکه عصبی با استفاده از ترکیب پردازنده مرکزی و گرافیکی (به‌وسیله اُپن.اِم.پی. و کودا) در مقایسه با پردازنده مرکزی و پردازنده گرافیکی به تنهایی، به ترتیب به تسریع‌های ۱۵ و ٤ برابری رسید. در این پیاده‌سازی، استخراج ویژگی روی پردازنده مرکزی چند هسته‌ای و ضرب ماتریسی روی پردازنده گرافیکی صورت گرفت [۵۸]. به‌طور کلی می‌توان گفت، در صورت برگزیدن بخش مناسبی از کد برای اجرا روی پردازنده گرافیکی، سربار نقل و انتقالات حافظه نسبت به تسریع به‌دست آمده ناچیز خواهد بود و اجرای یک راه‌حل موازی برای پیاده‌سازی الگوریتم‌های تکاملی با کمک پردازنده گرافیکی می‌تواند در مقایسه با پردازنده‌های سنتی به تسریعی تا ۱۰۰ برابر برسد [٦۳].

محاسبات یادگیری عمیق روی پردازنده گرافیکی به دو صورت انجام می‌شود [۳٦]: ۱) تقسیم کردن مجموعه داده و آموزش یک مدل واحد روی داده متفاوت (موازی‌سازی داده) و ۲) تقسیم کردن مدل و اجرای بخش‌های مختلف شبکه روی ابزارهای متفاوت با استفاده از یک مدل شبه خطلوله‌ای (موازی‌سازی مدل). امروزه تقریبا تمام گروه‌هایی که روی آموزش شبکه‌های عمیق کار می‌کنند، از پردازنده گرافیکی استفاده می‌کنند [۳۵]. شبکه‌های عصبی عمیق در صورتی‌که حجم محاسباتی متناسب با پردازنده گرافیکی داشته باشند، می‌توانند تسریع‌های چشمگیری را رقم بزنند. شناسایی علائم راهنمایی و رانندگی با پیاده‌سازی یک شبکه عصبی عمیق روی پردازنده گرافیکی به نرخ شناسایی بهتر از انسان، برابر ۹۹,٤٦٪ دست یافت [٦٤]. پیچیدگی محاسباتی شبکه عصبی بازگشتی، آموزش کارآمد این شبکه‌ها را دشوار ساخته بود. با ظهور پردازنده‌های گرافیکی، آموزش کارآمد شبکه‌های بزرگ عصبی بازگشتی میسر شده و تسریع حداقل ۲-۱۱ برابری در مقایسه با پردازنده مرکزی قابل دستیابی است [٦۵]. پیاده‌سازی آموزش و رده‌بندی شبکه‌های عصبی پیچشی روی پردازنده گرافیکی در مقایسه با پردازنده مرکزی به تسریع ۲ تا ۲٤ برابری، بسته به هم‌بندی[۲۳] شبکه، می‌رسد [٦٦]. به عنوان یک نمونه کاربردی از پیاده‌سازی شبکه عصبی پیچشی روی پردازنده گرافیکی، دسته‌بندی سه نوع چای با استفاده از شبکه عصبی پیچشی روی پردازنده گرافیکی را مطرح می‌کنیم که به دقت کلی ۹۸,۳۳٪ رسیده و در آن تسریع داده آموزشی و داده آزمایشی روی پردازنده گرافیکی نسبت به پردازنده مرکزی به ترتیب ۱۷۵ برابر و ۱۲۲ برابر است [٦۷].

تسریع‌های به‌دست آمده با کمک پردازنده گرافیکی در حوزه شبکه‌های عصبی قابل توجه است. با این‌حال، جهت به‌کارگیری موفق هر معماری موازی و زیرساخت توزیع-شده برای محاسبات تکاملی، لازم است موضوعاتی همچون زمان‌بندی، انتقال برنامه‌های کاربردی، همبندی ارتباطات، مدل‌ها و معماری‌های موازی جدید، بهینه‌سازی مدیریت حافظه و حافظه نهان، مصرف انرژی، چند رشته‌ای‌سازی همزمان و چندکارگی انحصاری ۲٤ مد نظر قرار داده شوند [٦۹]. به رغم تناسب ساختاری میان الگوی اِس.پی.اِم.دی. پردازنده گرافیکی و مشخصه‌های الگوریتم‌های تکاملی، چالش‌هایی نیز وجود دارد که نیازمند پیش‌بینی و اجتناب است. مثلا ماهیت تصادفی این الگوریتم‌ها می‌تواند به واگرایی و در نتیجه افت کارایی منتهی گردد [۷۰]. رفع این پیچیدگی‌ها، نیازمند ارتباطی دوسویه میان جامعه متخصصین پردازش موازی و محاسبات تکاملی است.

٤-۱ ابزارها

در خصوص محاسبات تکاملی، تلاش‌هایی برای تسهیل برنامه‌نویسی صورت گرفته است. ایزی[۲۵] چارچوبی‌ست

23- Topology

۲۴- Preemptive multitasking: سیستم چند وظیفه‌ای که در آن زمان‌بند قادر است موقتا وظیفه‌ای را متوقف کرده و بعدا مجددا به آن بازگردد.

۲۵-Easy Specification for Evolutionary Algorithms (EASEA): اولین نسخهٔ این

جدول ۲: پشتیبانی از پردازش موازی پردازنده گرافیکی در متلب

جعبه ابزار	توضیح
جعبه ابزار پردازش تصویر [۳۴، ۷۸]	حاوی توابع زیادی است که از پردازنده گرافیکی پشتیبانی می‌کنند. این کار به‌واسطه gpuArray میسر شده است.
جعبه ابزار یادگیری عمیق [۷۹]	چارچوبی برای طراحی و پیاده‌سازی شبکه‌های عصبی عمیق با الگوریتم‌ها، مدل‌های از پیش آموزش داده شده و اَپ‌ها. برای تسریع آموزش مجموعه داده‌های بزرگ، می‌توان محاسبات و داده را روی پردازنده‌های مرکزی و گرافیکی توزیع کرد. امکان مقیاس‌دهی۲ روی خوشه‌ها و ابرها نیز وجود دارد.
جعبه ابزار آمار و یادگیری ماشین [۳۴]	حاوی توابع زیادی است که از پردازنده گرافیکی پشتیبانی می‌کنند. این کار به‌واسطه gpuArray میسر شده است.
جعبه ابزار پردازش سیگنال [۳۴]	تعدادی از توابع این جعبه ابزار از پردازنده گرافیکی پشتیبانی می‌کنند. این کار به‌واسطه gpuArray میسر شده است.
جعبه ابزار طرف سوم مَت‌کانونِت۳ [۸۰]	یک پیاده‌سازی شبکه‌های عصبی پیچشی متلب برای کاربردهای بینایی ماشین است که بر سادگی و انعطاف‌پذیری تکیه دارد. این جعبه ابزار با پشتیبانی از محاسبات کارآمد روی پردازنده مرکزی و گرافیکی، امکان آموزش مدل‌های پیچیده روی مجموعه داده‌های بزرگ را فراهم نموده و نمونه‌سازی۴ پرسرعت معماری‌های جدید شبکه‌های عصبی پیچشی را میسر می‌سازد.

که به برنامه‌نویسان غیرمتخصص کمک می‌کند مسائل را با کمک محاسبات تکاملی بهینه‌سازی نمایند. این بستر نرم‌افزاری، امکان انتقال الگوریتم‌های تکاملی مختلف به پردازنده گرافیکی را فراهم می‌کند. همچنین با استفاده از مدل جزیره‌ای، امکان انتقال روی خوشه‌ای از ماشین‌های ناهمگن نیز وجود دارد [۷۱]. پیاده‌سازی یک الگوریتم تکاملی استاندارد با کمک این چارچوب به تسریعی تا ۳۰ برابر دست یافت. در واقع در این پیاده‌سازی، ارزیابی تمامی افراد مجموعه به‌صورت موازی روی پردازنده گرافیکی صورت گرفته است [۷۲]. اما برای این‌که بتوان از پردازنده گرافیکی به‌طور گسترده استفاده کرد، لازمست امکان کار با آن روی محیط‌های نرم‌افزاری رایج همچون متلب فراهم شده و کتابخانه‌های بروز و کاملی برای این کار در دسترس باشد.

پشتیبانی از پردازنده گرافیکی در جعبه ابزار پردازش موازی متلب در نظر گرفته شده است. از واسط‌های پردازنده گرافیکی متلب به نام‌های جَکِت و جی.پی.یو.مَت[۲۶] هم می‌توان بهره برد. سه جعبه ابزار شتابدهی متلب با استفاده از پردازنده گرافیکی- یعنی جَکِت، جی.پی.یو.مَت و جعبه ابزار محاسبات موازی متلب[۲۷]- با هدف شناسایی جعبه ابزار مناسب برای هر وظیفه مورد نظر، در [۳۲] مقایسه شده و مزایا و معایب آن‌ها بیان شده است. از آنجایی که جَکِت [۷۳] یک بسته تجاری طرف سوم بود و از نظر حقوقی با شرکت مَت‌وُرکس[۲۸] به توافق نرسید، مجوز آن دیگر فروخته نمی‌شود [۷۴]. جی.پی.یو.مَت نیز به‌علت عدم تأمین مالی تولیدکننده، از کودا۵ به بعد دیگر پشتیبانی نمی‌شود [۷۵]. اما جعبه ابزار محاسبات موازی متلب رو به توسعه بوده و برای محاسبات موازی یادگیری ماشین با پردازنده گرافیکی بسیار کاربردی است. نقات قوت و ضعف این جعبه ابزار در [۳۳] بررسی شده است. جعبه ابزار محاسبات موازی متلب به کاربران این امکان را می‌دهد تا با استفاده از توابع سطح بالای متلب و بدون استفاده از کودا، اِم.پی.آی. و یا اُپن.اِم.پی. به موازی‌سازی برنامه‌های خود بپردازند [۷۶]. به‌علاوه، با فایل‌های سی‌مِکس[۲۹] می‌توان از کتابخانه‌های کارآمد کودا (نظیر cuSPARSE،cuRAND،cuFFT،cuBLAS و Thrust) و سخت‌افزار گرافیکی نیز بهره حداکثری برد [۳۳]. امکان دیگری که برای پردازنده گرافیکی در متلب در نظر گرفته شده است، جی.پی.یو.کودِر[۳۰] است. جی.پی.یو.کودِر، از روی کد متلب برای یادگیری عمیق، بینایی نهفته[۳۱] و سیستم‌های

زبان، با هدف فراهم آوردن یک بن‌سازه نرم‌افزاری برای برنامه‌نویسان غیرمتخصص ارائه شد تا به کمک آن بتوانند از الگوریتم‌های تکاملی برای بهینه‌سازی مسائل کاربردی خود استفاده نمایند.

26-GPUmat & jacket
27-PCT: Parallel Computing Toolbox

28-Mathworks

۲۹-c-mex: توابع متلب نوشته شده با C/C++

30-GPU Coder
31- Embedded vision

جدول ۳: نتایج کلی حاصل از بررسی پنج ابزار یادگیری عمیق برای سه نوع شبکه عصبی روی پردازنده مرکزی و گرافیکی

نتیجه [۸۱]	توضیح
عدم افزایش کارایی با افزایش تعداد هسته‌های پردازنده مرکزی	کارایی اجرا با ۱۶ هسته پردازنده مرکزی فقط کمی بهتر از ۸ یا ۴ هسته است.
کارآمدی بسیار بهتر پردازنده گرافیکی نسبت به پردازنده مرکزی	همه ابزارها در صورت استفاده از پردازنده گرافیکی به تسریع چشمگیری دست می‌یابند
امکان استفاده از چند پردازنده گرافیکی	با هدف دستیابی به توان عملیاتی بسیار بالاتر و تسریع در سرعت همگرایی با موازی‌سازی داده در طول آموزش
مقیاس‌پذیری نسبتا بهتر تنسِر فلو در مقایسه با سایر ابزارها	
کارایی بهتر اجرای شبکه‌های عصبی کاملا متصل روی تک پردازنده گرافیکی با کَفِی، سی‌اِن‌تی‌کا و تورچ نسبت به اِم‌اِکس‌نت و نتسرفلو	
تناسب عالی اِم‌اِکس‌نت برای شبکه‌های عصبی پیچشی بزرگ	
کارایی خوب کَفِی و سی‌اِن‌تی‌کا برای شبکه‌های عصبی پیچشی بزرگ و حتی کوچک	
کارآمدی زمانی عالی سی‌اِن‌تی‌کا برای شبکه‌های عصبی بازگشتی	کارآمدی زمانی ۵-۱۰ برابر بهتر از سایر ابزارها
مقیاس‌دهی خوب سی‌اِن‌تی‌کا به شبکه‌های عصبی کاملا متصل	
مقیاس‌دهی عالی ام‌اِکس‌نت و تورچ به شبکه‌های عصبی پیچشی	

خودکار می‌تواند کد بهینهٔ کودا بسازد. کد ساخته شده، کتابخانه‌های بهینهٔ کودا را فراخوانی می‌نماید [۷۷]. جعبه ابزارهایی که از پردازش موازی پردازنده گرافیکی در متلب پشتیبانی می‌کنند در جدول (۲) معرفی شده‌اند.

به‌علت تنوع ابزارهای نرم‌افزاری و بسترهای سخت‌افزای، انتخاب نرم‌افزار و سخت‌افزار مناسب برای اجرای کارهای یادگیری عمیق مورد نظر می‌تواند برای کاربران دشوار باشد. با اشاره به این نکته، شی و همکاران [۸۱] به بررسی مجموعه‌ای از ابزارهای یادگیری عمیقِ شتاب‌یافته با پردازنده گرافیکی پرداختند. آن‌ها نتیجه گرفته‌اند که ابزارهای مختلف، ویژگی‌ها و کاراییِ زمانِ اجرایِ متفاوتی را حین آموزش شبکه‌های عمیق متفاوت بر بسترهای سخت‌افزاری متفاوت از خود نشان می‌دهند. ابزارهای تورچ، تنسِرفلو، اِم‌اِکس‌نت، سی‌اِن‌تی‌کا و کَفِی[۳۲] با استفاده از سه نوع اصلی شبکه یادگیری عمیق (شبکه‌های عصبی کاملا متصل، شبکه‌های عصبی پیچشی و شبکه‌های عصبی بازگشتی) تحلیل و ارزیابی شدند. نتایج کلی حاصل از این بررسی، در قالب جدول (۳) خلاصه شده و جزئیات بیشتر شامل جداول مقایسه‌ای حاصل از محک‌زنی‌های[۳۳] انجام شده در [۸۱] قابل مشاهده است. مقایسه پنج چارچوب یادگیری عمیق (تورچ، تینو، تنسِرفلو[۳۴]، نییان و کَفِی) از سه جنبه توسعه‌پذیری[۳۵]، بهره‌برداری سخت‌افزاری[۳۶] و سرعت روی پردازنده مرکزی و گرافیکی [۳۵] نیز در جدول (٤) آمده است. مروری جامع از نرم‌افزارها، کتابخانه‌ها و چارچوب‌های جدید هوش مصنوعی و مزایا و معایب آن‌ها در [۳۹] ارائه شده است. در این مقاله به ۱) چارچوب‌ها و کتابخانه‌های یادگیری ماشین بدون پشتیبانی سخت‌افزاری ویژه، ۲) چارچوب‌ها و کتابخانه‌های یادگیری عمیق با پشتیبانی از پردازنده گرافیکی و ۳) چارچوب‌ها و کتابخانه‌های یادگیری عمیق با پشتیبانی از مَپ‌ریدوس[۳۷] پرداخته شده است.

٤-۲ محدودیت‌ها و الزامات

طی چند سال گذشته، تعداد پیاده‌سازی‌های الگوریتم‌های

32- Caffe, CNTK, MXNet, TensorFlow & Torch
33- Benchmarking

۳۴- یکی دیگر از کتابخانه‌های رایج شبکه‌های عصبی، Keras است. این کتابخانه متن‌بازِ نوشته شده با پایتون، به‌عنوان یک کتابخانهٔ پوششی روی تنسرفلو یا تینو اجرا می‌شود. بنابراین جداگانه به آن پرداخته نمی‌پردازیم. کراس از پردازنده گرافیکی پشتیبانی کرده و قابلیت اجرا روی چند پردازنده گرافیکی را نیز دارد.

۳۵- Extensibility: قابلیت لحاظ کردن انواع مختلف معماری‌های یادگیری عمیق، انواع مختلف رویه‌های آموزش و الگوریتم‌های پیچشی مختلف.

۳۶-Hardware utilization: کارآمدی در بهره‌برداری از منابع سخت‌افزاری

۳۷- Map Reduce: یک مدل برنامه‌سازی برای پردازش و ایجاد مجموعه داده‌های بزرگ با یک الگوریتم موازی توزیع شده

جدول ۴: مقایسه پنج چارچوب یادگیری عمیق [۳۵]

مشخصه	Caffe	Neon	TensorFlow	Theano	Torch
هسته	سی++	پایتون	سی++	پایتون	Lua
پردازنده مرکزی	✔	✔	✔	✔	✔
پردازنده مرکزی چندرشته‌ای	Blas۵	فقط بارکننده داده	Eigen۶	Blas, conv۲D, limited OpenMP	استفاده گسترده
پردازنده گرافیکی	✔	زیرساخت سفارشی‌سازی شده انویدیا[۷]	✔	✔	✔
چند پردازنده گرافیکی	فقط داده موازی	✔	انعطاف‌پذیرترین	نسخه آزمایشی	✔
انویدیا cuDNN۸	✔	✕	✔	✔	✔
اعمال سریع روی مدل‌های استاندارد	سادهترین	✔	✔	با کمک کتابخانه‌های فرعی	✔
محاسبه خودکار گرادیان	✔	پشتیبانی از Op-Tree[۹]	✔	انعطاف‌پذیرترین حتی روی حلقه‌ها	✔

یادگیری ماشین روی پردازنده گرافیکی رشد چشمگیری داشته است اما بیشتر این پیاده‌سازی‌ها به‌صورت عمومی به اشتراک گذاشته نشده‌اند و این موضوع موجب دشواری تکرار پیاده‌سازی‌ها و به‌کارگیری‌شان در وظایف جدید یادگیری ماشین شده است [۸۲].

به‌علاوه، اجرای کارآمد مدل‌های بزرگ‌تر شبکه‌های عصبی عمیق، چالش‌برانگیز است. آموزش یک شبکه عمیق، ذاتا از نظر محاسباتی حجیم بوده و به حافظه زیادی نیاز دارد. با توسعه کاربرد شبکه‌های عصبی عمیق، تمایل به سمت شبکه‌های عمیق‌تر پیش رفته که به تعداد پارامترهای بیشتر و اندازه مدل بزرگ‌تری هم نیاز دارند. هرچه در شبکه عمیق‌تر یا گسترده‌تر پیش برویم، حافظه بیشتری هم نیاز داریم. به دلیل ناکافی بودن حافظه سریع برتراشه برای این کار باید از چند پردازنده گرافیکی استفاده نمود. اولین نتیجه این روش، پهنای باند کمتری است که در اختیار تعاملاتِ میان پردازنده‌هایِ حافظه قرار دارد و می‌تواند به قیمت افت کارایی تمام شود. با اشاره به این محدودیت، یک زمانبند پویای حافظه زمان اجرا برای پردازنده گرافیکی به نام سوپرنرونز[۳۸] ارائه شده که آموزش شبکه را ورای ظرفیت دی‌رَم پردازنده گرافیکی میسر سازد [۸۳]. به‌علاوه، استفاده ترکیبی از توانمندی‌های پردازنده گرافیکی و چارچوب مَپ‌ریدوس می‌تواند بسیاری از این محدودیت‌های کارایی را مرتفع ساخته و تمرکز پژوهش‌های آتی باشد [۳۶]. موانع دیگر مقیاس‌پذیری مدل‌های یادگیری عمیق روی چند پردازنده گرافیکی عبارتند از عدم پشتیبانی کتابخانه مورد استفاده از ارتباطات میان پردازنده‌های گرافیکی و ضرورت بازنویسی کد برای بهره‌برداری از چند پردازنده گرافیکی. با اشاره به قابل چشم‌پوشی نبودن سربار ارتباطاتِ میان‌پردازنده‌ای و لزوم بازنویسی قابل توجه کد کاربر، یک کتابخانه متن‌باز به نام هورُوُد[۳۹] برای حل این موانع پیشنهاد شده است [۸۴].

محدودیت دیگر شبکه‌های عصبی عمیق، انرژی‌بر بودن آن‌هاست. تلاش پردازنده‌های مرکزی و گرافیکی برای فراهم نمودن انعطاف‌پذیری، منجر به ناکارآمدی‌شان از حیث مصرف انرژی شده است. گرایش کنونی، به سمت طراحی هم‌پردازنده‌های اختصاصی با هدف به‌کارگیری در کنار پردازنده‌های عمومی است. در این رویکرد، پردازنده‌های عمومی و شتاب‌دهنده‌ها به‌صورت ترکیبی کار می‌کنند [۳۸]. بنابراین شتاب‌دهنده‌های سخت‌افزاری از حیث کاهش مصرف انرژی قابل توجه هستند. بررسی رشد سخت‌افزاری اِف.پی.جی.اِی. شرکت اینتل و تمایلات الگوریتمی شبکه‌های عصبی مصنوعی به این نتیجه می‌رسد که اِف.پی.جی.اِی. برای نسل‌های بعدی شبکه عصبی عمیق که تمایل به استفاده از انواع داده فشرده، بهره‌برداری از خلوتی (مقادیر صفر

38-SuperNeurons

39-Horovod

جدول ۵: خلاصه محدودیت‌ها و الزامات

پیشنهاد	ضرورت
به اشتراک‌گذاری پیاده‌سازی‌های الگوریتم‌های یادگیری ماشین روی پردازنده گرافیکی [۸۲]	تکرارپذیری پیاده‌سازی‌ها و امکان به‌کارگیری‌شان در وظایف جدید یادگیری ماشین
مدیریت و بهینه‌سازی استفاده از پهنای باند تعاملاتِ میان پردازنده‌ایِ حافظه [۸۳]	مقابله با افت کارایی ناشی از تعاملات حافظه
پشتیبانی کتابخانه‌ها از ارتباطات میان پردازنده‌های گرافیکی [۸۴]	مقیاس‌پذیری مدل‌های یادگیری عمیق روی چند پردازنده گرافیکی با هدف بهبود کارایی
رفع ضرورت بازنویسی کد توسط کاربر برای بهره‌برداری از چند پردازنده گرافیکی [۸۴]	
ترکیب مَپ‌ریدوس و پردازنده گرافیکی [۳۶, ۳۹]	
کارآمدی مصرف انرژی: مقایسه آخرین معماری پردازنده‌های گرافیکی (وُلتا و تورینگ) با سایر شتاب‌دهنده‌های سخت‌افزاری نظیر اِف‌پی‌جی‌اِی و اِی‌سیک از حیث مصرف انرژی	مقایسه‌هایی که میان پردازنده مرکزی، پردازنده گرافیکی و شتاب‌دهنده‌های سخت‌افزاری صورت گرفته همگی بر اساس پردازنده‌های گرافیکی قبل از وُلتا انجام شده است

برای وزن‌ها) و در نتیجه موازی‌سازی نامنظم دارند، بهتر از پردازنده گرافیکی عمل خواهد کرد [۳۷]. شتاب‌دهنده‌های مبتنی بر اِی.سیک نیز بهبود قابل توجهی از حیث انرژی ارائه داده‌اند، ضمن این‌که کارایی مطلوبی هم دارند [۸۵]. بنابراین به‌نظر می‌رسد رویکرد آینده، به سمت بهره‌برداری از سخت‌افزار اختصاصی هوش مصنوعی و یادگیری ماشین باشد. مقایسه‌هایی که میان پردازنده مرکزی، پردازنده گرافیکی و شتاب‌دهنده‌های سخت‌افزاری صورت گرفته [۳۸] همگی بر اساس پردازنده‌های گرافیکی قبل از وُلتا انجام شده است. این در حالیست که پردازنده‌های گرافیکی با معرفی ریز معماری وُلتا انویدیا [۸۶]، همین رویکرد سخت‌افزار اختصاصی را در پیش گرفته است. بنابراین برای درکی واقع‌بینانه از جایگاه پردازنده گرافیکی در حوزه یادگیری عمیق، لازمست آخرین معماری این پردازنده‌ها (وُلتا و تورینگ) با سایر شتاب‌دهنده‌های سخت‌افزاری مقایسه شود. خلاصه‌ای از این پیشنهادها، محدودیت‌ها و الزامات در جدول (۵) نشان داده شده است.

جمع‌بندی و نتیجه‌گیری

در این مقاله به روند رشد پردازنده‌ها پرداخته و دیدیم که چطور پردازنده گرافیکی به سرعت تبدیل به بستری رایج برای پردازش موازی شد و تسریع‌های چشمگیری را رقم زد. قیمت مناسب، توان پردازشی بالا، انعاف‌پذیری و ابزارهای برنامه‌نویسی رو به توسعه، موجب گسترش حوزه‌های کاربرد این پردازنده‌ها شده است. از میان طیف وسیع شاخه‌های علمی و صنعتی که تاکنون از توان محاسباتی پردازنده‌های گرافیکی بهره برده‌اند، در این مقاله به‌طور خاص به حوزهٔ شبکه‌های عصبی و یادگیری عمیق پرداختیم. حوزه یادگیری ماشین از همان ابتدا مورد توجه برنامه‌نویسان پردازنده گرافیکی قرار گرفته است و تا امروز، ابزارها و کتابخانه‌های این حوزه روی پردازنده گرافیکی رشد خوبی داشته‌اند به‌طوری که در مطلب هم به راحتی در دسترس قرار گرفته‌اند. با معرفی زیرمعماری ولتا انویدیا، پردازنده‌های گرافیکی به‌صورت تخصصی وارد حوزه هوش مصنوعی و یادگیری ماشین شد. با وجود این رشد و نتایج قابل توجه، هنوز زوایایی هستند که نیازمند توجه هستند. توسعه بیشتر کتابخانه‌های متن باز، بهینگی مصرف‌انرژی و محدودیت حافظه و پهنای باند از آن جمله‌اند. ضمن این‌که مقایسه ریزمعماری اختصاصی پردازنده گرافیکی برای یادگیری ماشین با سایر شتاب‌دهنده‌های سخت‌افزاری نظیر اِف.پی.جی.اِی. و اِی.سیک هنوز انجام نشده است. بررسی فرصت به‌کارگیری سیستم‌های ترکیبی با هدف بهره‌برداری از مزایا و توانمندی‌های هر بستر سخت‌افزاری بنا به تناسب وظیفه مورد نظر هم می‌تواند مفید باشد.

مراجع

[1] H. Sutter and J. Larus, "Software and the Concurrency Revolution," Queue, vol. 3, no. 7, pp. 54-62, 2005.

[2] D. B. Kirk and W.-m. W. Hwu, Programming Massively Parallel Processors: A Hands-on Approach. Morgan Kaufmann Publishers Inc., 2010, p. 280.
[3] H. Esmaeilzadeh, E. Blem, R. S. Amant, K. Sankaralingam, and D. Burger, "Dark Silicon and the End of Multicore Scaling," IEEE Micro, vol. 32, no. 3, pp. 122-134, 2012.
[4] J. Nickolls and W. J. Dally, "The GPU Computing Era," IEEE Micro, vol. 30, no. 2, pp. 56-69, 2010.
[5] Nvidia. (2019). GPU Application Catalog [Online]. Available:https://www.nvidia.com/content/dam/en-zz/Solutions/Data-Center/tesla-product-literature/gpu-applications-catalog.pdf.
[6]M. Ekman, F. Warg, and J. Nilsson, "An in-depth look at computer performance growth," SIGARCH Comput. Archit. News, vol. 33, no. 1, pp. 144-147, 2005.
[7]J. D. Owens et al., "A Survey of General-Purpose Computation on Graphics Hardware," Computer Graphics Forum, vol. 26, no. 1, pp. 80-113, 2007.
[8]S. W. Keckler, W. J. Dally, B. Khailany, M. Garland, and D. Glasco, "GPUs and the Future of Parallel Computing," IEEE Micro, vol. 31, no. 5, pp. 7-17, 2011.
[9]G. Colin de Verdière, "Introduction to GPGPU, a hardware and software background," Comptes Rendus Mécanique, vol. 339, no. 2, pp. 78-89, 2011/02/01/ 2011.
[10]A. R. Brodtkorb, T. R. Hagen, and M. L. Sætra, "Graphics processing unit (GPU) programming strategies and trends in GPU computing," Journal of Parallel and Distributed Computing, vol. 73, no. 1, pp. 4-13, 2013/01/01/ 2013.
[11]J. Owens, "Streaming Architectures and Technology Trends," in GPU Gems 2: Programming Techniques for High-Performance Graphics and General-Purpose Computation (Gpu Gems), M. Pharr and R. Fernando, Eds.: Addison-Wesley Professional, 2005.
[12]Nvidia. (2018). Unlimited Savings with Volta Infographic [Online]. Available: http://images.nvidia.com/content/volta-architecture/pdf/more-you-buy-infographic-r6-web.pdf.
[13]Nvidia. (2019). Cuda C Programming Guide [Online]. Available: https://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html.
[14]J. D. Owens, M. Houston, D. Luebke, S. Green, J. E. Stone, and J. C. Phillips, "GPU Computing," Proceedings of the IEEE, vol. 96, no. 5, pp. 879-899, 2008.
[15]A. Lefohn, J. Kniss, and J. Owens, "Implementing efficient parallel data structures on GPUs," in GPU Gems 2: Programming Techniques for High-Performance Graphics and General-Purpose Computation, M. Pharr and R. Fernando, Eds., 2005.
[16]R. Farber, CUDA Application Design and Development. Morgan Kaufmann Publishers Inc., 2012, p. 336.
[17]Nvidia. (2018). Accelerated Computing and the Democratization of Supercomputing [Online]. Available: https://www.nvidia.com/content/dam/en-zz/Solutions/Data-Center/tesla-product-literature/sc18-tesla-democratization-tech-overview-r4-web.pdf.
[18]Nvidia. (2012). NVIDIA® TESLA® KEPLER GPU COMPUTING ACCELERATORS [Online]. Available: https://www.nvidia.com/content/tesla/pdf/nv-ds-teslak-family-jul2012-lr.pdf.
[19]Nvidia. (2014, 03/10/2019). 5 Things You Should Know About the New Maxwell GPU Architecture [Online]. Available: https://devblogs.nvidia.com/5-things-you-should-know-about-new-maxwell-gpu-architecture/.
[20]Nvidia. (2014, 03/10/2019). NVIDIA Updates GPU Roadmap; Announces Pascal [Online]. Available: https://blogs.nvidia.com/blog/2014/03/25/gpu-roadmap-pascal/.
[21]Nvidia. (2019, 03/10/2019). Volta GPU Architecture [Online]. Available: https://www.nvidia.com/en-us/data-center/volta-gpu-architecture/.
[22]Nvidia. (2019). Nvidia Nsight Eclipse Edition [Online]. Available: https://developer.nvidia.com/nsight-eclipse-edition.
[23]A. Eklund, P. Dufort, D. Forsberg, and S. M. LaConte, "Medical image processing on the GPU – Past, present and future," Medical Image Analysis, vol. 17, no. 8, pp. 1073-1094, 2013/12/01/ 2013.
[24]Nvidia. (2019, 03/10/2019). GPU Accelerated Libraries [Online]. Available: https://developer.nvidia.com/gpu-accelerated-libraries.
[25]M. Boyer, K. Skadron, and W. Weimer, "Automated Dynamic Analysis of CUDA Programs," 2008.
[26]K. Kothapalli, R. Mukherjee, M. S. Rehman, S. Patidar, P. J. Narayanan, and K. Srinathan, "A performance prediction model for the CUDA GPGPU platform," in 2009 International Conference on High Performance Computing (HiPC), 2009, pp. 463-472.
[27]J. Sim, A. Dasgupta, H. Kim, and R. Vuduc, "A performance analysis framework for identifying potential benefits in GPGPU applications," presented at the Proceedings of the 17th ACM SIGPLAN symposium on Principles and Practice of Parallel Programming, New Orleans, Louisiana, USA, 2012.
[28]C. Gregg and K. Hazelwood, "Where is the data? Why you cannot debate CPU vs. GPU performance without the answer," in (IEEE ISPASS) IEEE International Symposium on Performance Analysis of Systems and Software, 2011, pp. 134-144.
[29]J. V. F. Lima, T. Gautier, N. Maillard, and V. Danjean, "Exploiting Concurrent GPU Operations for Efficient Work Stealing on Multi-GPUs," in 2012 IEEE 24th International Symposium on Computer Architecture and High Performance Computing, 2012, pp. 75-82.
[30]Y. Torres, A. Gonzalez-Escribano, and D. R. Llanos, "Using Fermi Architecture Knowledge to Speed up CUDA and OpenCL Programs," in 2012 IEEE 10th International Symposium on Parallel and Distributed Processing with Applications, 2012, pp. 617-624.
[31] R. J. Meuth and D. Wunsch, A Survey of Neural Computation on Graphics Processing Hardware. 2007, pp. 524-527.
[32] Z. Baida, X. Shuai, Z. Feng, B. Yuan, and H. Linqi, "Accelerating MatLab code using GPU: A review of tools and strategies," in 2011 2nd International Conference on Artificial Intelligence, Management Science and Electronic Commerce (AIMSEC), 2011, pp. 1875-1878.

in Computational Science – ICCS 2006, Berlin, Heidelberg, 2006: Springer Berlin Heidelberg, pp. 236-243.
[48] K. Chellapilla, S. Puri, and P. Simard, "High performance convolutional neural networks for document processing," 2006.
[49] D. Steinkraus, I. Buck, and P. Y. Simard, "Using GPUs for machine learning algorithms," in Eighth International Conference on Document Analysis and Recognition (ICDAR'05), 2005, pp. 1115-1120 Vol. 2.
[50] W. Man-Leung, W. Tien-Tsin, and F. Ka-Ling, "Parallel evolutionary algorithms on graphics processing unit," in 2005 IEEE Congress on Evolutionary Computation, 2005, vol. 3, pp. 2286-2293 Vol. 3.
[51] Q. Yu, C. Chen, and Z. Pan, "Parallel Genetic Algorithms on Programmable Graphics Hardware," in Advances in Natural Computation, Berlin, Heidelberg, 2005: Springer Berlin Heidelberg, pp. 1051-1059.
[52] A. Campbell, E. Berglund, and A. Streit, "Graphics hardware implementation of the parameter-less self-organising map," presented at the Proceedings of the 6th international conference on Intelligent Data Engineering and Automated Learning, Brisbane, Australia, 2005.
[53] L. Zhongwen, L. Hongzhi, and W. Xincai, "Artificial neural network computation on graphic process unit," in Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005., 2005, vol. 1, pp. 622-626 vol. 1.
[54] K.-S. Oh and K. Jung, "GPU implementation of neural networks," Pattern Recognition, vol. 37, no. 6, pp. 1311-1314, 2004/06/01/ 2004.
[55] A. Guzhva, S. Dolenko, and I. Persiantsev, "Multifold Acceleration of Neural Network Computations Using GPU," in Artificial Neural Networks – ICANN 2009, Berlin, Heidelberg, 2009: Springer Berlin Heidelberg, pp. 373-380.
[56] R. Raina, A. Madhavan, and A. Y. Ng, "Large-scale deep unsupervised learning using graphics processors," presented at the Proceedings of the 26th Annual International Conference on Machine Learning, Montreal, Quebec, Canada, 2009.
[57] A. Brandstetter and A. Artusi, "Radial Basis Function Networks GPU-Based Implementation," Trans. Neur. Netw., vol. 19, no. 12, pp. 2150-2154, 2008.
[58] H. Jang, A. Park, and K. Jung, "Neural Network Implementation Using CUDA and OpenMP," in 2008 Digital Image Computing: Techniques and Applications, 2008, pp. 155-161.
[59] T. Sharp, "Implementing Decision Trees and Forests on a GPU," in Computer Vision – ECCV 2008, Berlin, Heidelberg, 2008: Springer Berlin Heidelberg, pp. 595-608.
[60] P. Trebatický and J. Pospíchal, "Neural Network Training with Extended Kalman Filter Using Graphics Processing Unit," in Artificial Neural Networks - ICANN 2008, Berlin, Heidelberg, 2008: Springer Berlin Heidelberg, pp. 198-207.
[61] S. A. A. Shalom, M. Dash, and M. Tue, "Efficient K-Means Clustering Using Accelerated Graphics Processors," in Data Warehousing and Knowledge Discovery, Berlin, Heidelberg, 2008: Springer Berlin Heidelberg, pp. 166-175.
[62] M. Martínez-Zarzuela, F. J. Díaz Pernas, J. F. Díez
[33] J. W. Suh and Y. Kim, Accelerating MATLAB with GPU Computing: A Primer with Examples. Morgan Kaufmann Publishers Inc., 2013, p. 258.
[34] N. Ploskas and N. Samaras, "Chapter 5 - GPU programming on MATLAB toolboxes," in GPU Programming in MATLAB, N. Ploskas and N. Samaras, Eds. Boston: Morgan Kaufmann, 2016, pp. 109-170.
[35] S. Bahrampour, N. Ramakrishnan, L. Schott, and M. Shah, "Comparative Study of Caffe, Neon, Theano, and Torch for Deep Learning," CoRR, 2015.
[36] A. Cano, "A survey on graphic processing unit computing for large-scale data mining," Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, vol. 8, no. 1, p. e1232, 2018/01/01 2018.
[37] E. Nurvitadhi et al., "Can FPGAs Beat GPUs in Accelerating Next-Generation Deep Neural Networks?," presented at the Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, Monterey, California, USA, 2017.
[38] Z. Li, Y. Wang, T. Zhi, and T. Chen, "A survey of neural network accelerators," Frontiers of Computer Science, vol. 11, no. 5, pp. 746-761, 2017/10/01 2017.
[39] G. Nguyen et al., "Machine Learning and Deep Learning frameworks and libraries for large-scale data mining: a survey," Artificial Intelligence Review, vol. 52, no. 1, pp. 77-124, 2019/06/01 2019.
[40] N. Lopes and B. Ribeiro, "GPU implementation of the multiple back-propagation algorithm," presented at the Proceedings of the 10th international conference on Intelligent data engineering and automated learning, Burgos, Spain, 2009.
[41] J. M. Nageswaran, N. Dutt, J. L. Krichmar, A. Nicolau, and A. V. Veidenbaum, "A configurable simulation environment for the efficient simulation of large-scale spiking neural networks on graphics processors," Neural Networks, vol. 22, no. 5, pp. 791-800, 2009/07/01/ 2009.
[42] V. Garcia, E. Debreuve, and M. Barlaud, "Fast k nearest neighbor search using GPU," in 2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, 2008, pp. 1-6.
[43] W. B. Langdon and W. Banzhaf, "A SIMD Interpreter for Genetic Programming on GPU Graphics Cards," in Genetic Programming, Berlin, Heidelberg, 2008: Springer Berlin Heidelberg, pp. 73-85.
[44] B. Catanzaro, N. Sundaram, and K. Keutzer, "Fast support vector machine training and classification on graphics processors," presented at the Proceedings of the 25th international conference on Machine learning, Helsinki, Finland, 2008.
[45] A. Brunton, S. Chang, and G. Roth, "Belief Propagation on the GPU for Stereo Vision," in The 3rd Canadian Conference on Computer and Robot Vision (CRV'06), 2006, pp. 76-76.
[46] Q. Yang, L. Wang, R. Yang, S. Wang, M. Liao, and D. Nister, "Real-time Global Stereo Matching Using Hierarchical Belief Propagation," in BMVC, 2006, vol. 6: Citeseer, pp. 989-998.
[47] F. Bernhard and R. Keriven, "Spiking Neurons on GPUs,"

ing Toolbox," in GPU Programming in MATLAB, N. Ploskas and N. Samaras, Eds. Boston: Morgan Kaufmann, 2016, pp. 37-70.
[77] Mathworks. (08/06/2019). GPU Coder [Online]. Available: https://www.mathworks.com/products/gpu-coder.html.
[78] Mathworks. (08/06/2019). Image Processing Toolbox [Online]. Available: https://www.mathworks.com/products/image.html.
[79] Mathworks. (08/06/2019). Deep Learning Toolbox [Online]. Available: https://www.mathworks.com/products/deep-learning.html.
[80] A. Vedaldi and K. Lenc, "MatConvNet: Convolutional Neural Networks for MATLAB," presented at the Proceedings of the 23rd ACM international conference on Multimedia, Brisbane, Australia, 2015.
[81] S. Shi, Q. Wang, P. Xu, and X. Chu, "Benchmarking State-of-the-Art Deep Learning Software Tools," in 2016 7th International Conference on Cloud Computing and Big Data (CCBD), 2016, pp. 99-104.
[82] S. Hasan, S. M. Shamsuddin, and N. Lopes, "Soft Computing Methods for Big Data Problems," in GPU Computing and Applications, Y. Cai and S. See, Eds. Singapore: Springer Singapore, 2015, pp. 235-247.
[83] L. Wang et al., "Superneurons: dynamic GPU memory management for training deep neural networks," presented at the Proceedings of the 23rd ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, Vienna, Austria, 2018.
[84] A. Sergeev and M. D. Balso, "Horovod: fast and easy distributed deep learning in TensorFlow," CoRR, vol. abs/1802.05799, 2018.
[85] S. Han et al., "EIE: efficient inference engine on compressed deep neural network," presented at the Proceedings of the 43rd International Symposium on Computer Architecture, Seoul, Republic of Korea, 2016.
[86] Nvidia. (2018). Tesla V100 Performance Guide: Deep Learning and HPC Applications [Online]. Available: http://images.nvidia.com/content/pdf/volta-marketing-v100-performance-guide-us-r6-web.pdf.
Higuera, and M. A. Rodríguez, "Fuzzy ART Neural Network Parallel Computing on the GPU," in Computational and Ambient Intelligence, Berlin, Heidelberg, 2007: Springer Berlin Heidelberg, pp. 463-470.
[63] O. Maitre, N. Lachiche, P. Clauss, L. Baumes, A. Corma, and P. Collet, "Efficient Parallel Implementation of Evolutionary Algorithms on GPGPU Cards," in Euro-Par 2009 Parallel Processing, Berlin, Heidelberg, 2009: Springer Berlin Heidelberg, pp. 974-985.
[64] D. Cireşan, U. Meier, J. Masci, and J. Schmidhuber, "Multi-column deep neural network for traffic sign classification," Neural Networks, vol. 32, pp. 333-338, 2012/08/01/ 2012.
[65] B. Li et al., "Large scale recurrent neural network on GPU," in 2014 International Joint Conference on Neural Networks (IJCNN), 2014, pp. 4062-4069.
[66] D. Strigl, K. Kofler, and S. Podlipnig, "Performance and Scalability of GPU-Based Convolutional Neural Networks," in 2010 18th Euromicro Conference on Parallel, Distributed and Network-based Processing, 2010, pp. 317-324.
[67] Y.-D. Zhang, K. Muhammad, and C. Tang, "Twelve-layer deep convolutional neural network with stochastic pooling for tea category classification on GPU platform," Multimedia Tools and Applications, vol. 77, no. 17, pp. 22821-22839, 2018/09/01 2018.
[68] N. Lopes, B. Ribeiro, and R. Quintas, "GPUMLib: A new Library to combine Machine Learning algorithms with Graphics Processing Units," in 2010 10th International Conference on Hybrid Intelligent Systems, 2010, pp. 229-232.
[69] EvoStar. (2018, 03/10/2019). EvoPAR 2018: Parallel Architectures and Distributed Infrastructures [Online]. Available: http://www.evostar.org/2018/cfp_evopar.php.
[70] S. Tsutsui and P. Collet, Massively Parallel Evolutionary Computation on GPGPUs. 2013.
[71] O. Maitre, F. Krüger, S. Querry, N. Lachiche, and P. Collet, "EASEA: specification and execution of evolutionary algorithms on GPGPU," Soft Computing, vol. 16, no. 2, pp. 261-279, 2012/02/01 2012.
[72] O. Maitre, L. A. Baumes, N. Lachiche, A. Corma, and P. Collet, "Coarse grain parallelization of evolutionary algorithms on GPGPU cards with EASEA," presented at the Proceedings of the 11th Annual conference on Genetic and evolutionary computation, Montreal, Québec, Canada, 2009.
[73] T. Larsen, G. Pryor, and J. Malcolm, "Chapter 28 - Jacket: GPU Powered MATLAB Acceleration," in GPU Computing Gems Jade Edition, W.-m. W. Hwu, Ed. Boston: Morgan Kaufmann, 2012, pp. 387-398.
[74] Mathworks. (2013, 08/07/2019). Will AccelerEyes Jacket functionality be implemented into MATLAB? [Online]. Available: https://www.mathworks.com/matlabcentral/answers/83853-will-accelereyes-jacket-functionality-be-implemented-into-matlab.
[75] G.-y. Group. (2014, 08/07/2019). GPUmat [Online]. Available: https://sourceforge.net/p/gpumat/discussion/help/thread/cba227c4/?limit=25#c785.
[76] N. Ploskas and N. Samaras, "Chapter 3 - Parallel Comput-