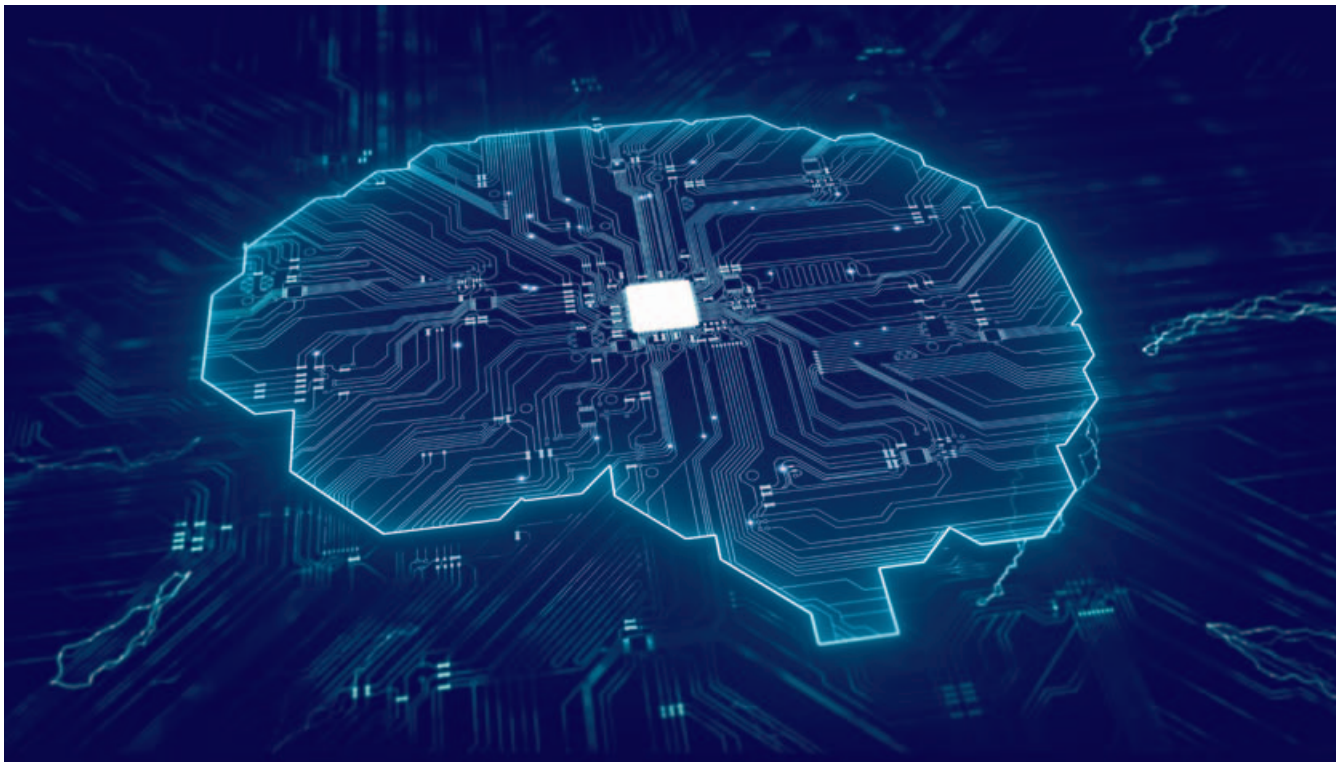

Bild: Lee | Adobe Stock

Neuromorphe Hardware

Neuronale Netze haben den Sprung von der Forschung in die Anwendung geschafft. Sie zeichnen sich durch spezielle Rechenoperationen aus, die durch dedizierte Hardware hinsichtlich Kosten, Geschwindigkeit und Energiebudget optimiert werden können. Ein Blick auf die verschiedenen Ansätze und den Entwicklungsstatus.

Johannes Leugering
Fraunhofer-Institut für Integrierte Schaltungen

Offizieller Medienpartner

embeddedworld2020 Exhibition&Conference ...it's a smarter world

APPROVED

Dieser Beitrag wurde auf der embedded world Conference 2020 präsentiert und von den Teilnehmern als besonders interessant bewertet.

Das letzte Jahrzehnt hat einen bemerkenswerten Wandel auf dem Gebiet der künstlichen Intelligenz mit sich gebracht. Seit 2012, als ein Deep-Neural-Network, das jetzt unter dem Namen AlexNet [1] bekannt ist, zum ersten Mal alle konkurrierenden Ansätze im Bereich der Bilderkennung ausstach, hat Deep Learning rasant an Popularität gewonnen – sowohl in der akademischen Welt als auch in der Industrie. Heute ist es zu einem allgegenwärtigen Werkzeug für viele Anwendungen geworden: Das Entsperren eines Mobiltelefons durch Gesichtserkennung [2], die Steuerung durch Sprache [3, 4] oder die Übersetzung einer Website [5] hängt wahrscheinlich von einem Deep-Neural-Network im Hintergrund ab, das die Hauptlast der Arbeit trägt. Am anderen Ende des Spektrums setzen leistungsfähige Server und Cluster tiefe neuronale Netze für die automatisierte Analyse großer medizinischer Datensätze [6], Wirtschaftsprognosen [7] oder epidemiologischer Vorhersagen [8] usw. ein, und es werden immer komplexere neuronale Netze entwickelt, um immer schwierigere Probleme zu bewältigen.

Infolge dieser »Deep Learning-Revolution« und während wir den allmählichen Niedergang des Mooreschen Gesetzes [9] beklagen, gibt es eine wachsende Nachfrage nach innovativen Hardware-Lösungen, um diese Entwicklung aufrechtzuerhalten - auch wenn wir uns den Grenzen der etablierten Technologien nähern. Insbesondere neuromorphe Hardware, die oft mit anderen Ansätzen unter den allgemeinen Oberbegriffen Next-Generation-Computing (NGC) oder Non-von-Neumann-Computing subsumiert wird, verspricht entscheidende Leistungsvorteile, die den Weg für eine weitere Markteinführung von maschinellem Lernen und künstlicher Intelligenz ebnen. Doch was genau ist neuromorphe Hardware, und wie funktioniert sie? Warum wird sie heute immer wichtiger, und wohin könnte sie uns in Zukunft führen? Diese Fragen werden

wir im Folgenden diskutieren, wobei wir mit ein wenig Hintergrundinformationen beginnen.

Neuromorphe Hardware ist darauf spezialisiert, neuronale Netze zu beschleunigen. Deshalb folgt hier ein Exkurs zum Verständnis neuronaler Netze, der hier aber nur sehr unvollständig und vereinfachend erfolgen kann. Glücklicherweise ist das mathematische Modell, das den meisten maschinenlernenden Anwendungen neuronaler Netze zugrunde liegt, relativ einfach.

Wie funktionieren neuronale Netze?

Im Kontext des maschinellen Lernens ist ein neuronales Netz eine Graphenstruktur, die aus Neuronen besteht, die durch Synapsen verbunden sind. Jede Synapse kann ihren Input, bei dem es sich um den Output eines Neurons oder einer externen Quelle handeln kann, mit einer Zahl, die als Gewicht der Synapse bezeichnet wird, multiplizieren und das Ergebnis an ein anderes Neuron übertragen. Jedes Neuron kombiniert linear alle Signale von seinen eingehenden synaptischen Verbindungen (skaliert durch ihr jeweiliges Gewicht) zu einem einzigen Signal. Dieses Signal wird dann nichtlinear transformiert, um den Ausgang des Neurons zu erzeugen.

Mathematisch ist der Ausgang $y_j(t)$ des Neurons j zum Zeitpunkt t daher eine Funktion der Eingangssignale $x_i(t)$ und hat die einfache Form

$$f(\sum_i w_{j,i}\, x_i(t) + b_j),$$

wobei $w_{j,i}$ die skalare Gewichtung einer synaptischen Verbindung vom Eingang x_i zum Neuron j darstellt und b_j ein neuronenspezifischer Offset- oder Bias-Term ist. Wir können alle Gewichte und Bias-Terme in einer einzigen Gewichtsmatrix W und dem Bias-Vektor b gruppieren, was zu der einfachen Matrixgleichung $y(t) = f(Wx(t) + b(t)$ führt. Für solche Matrixberechnungen stellen Multiply-Accumulate-Operationen (MACs) den Hauptrechenaufwand dar.

Eine deutliche Ausnahme davon bilden die Spiking-Neural-Networks (SNNs), die die Ausgabe eines Neurons stattdessen als Abfolge von kurzen, gleichartigen Impulsen (Spikes) codieren. (Wir diskutieren hier nur neuronale Netzwerkmodelle für das maschinelle Lernen, nicht bio-physikalisch genaue Modelle von Nervensystemen.) Diese Netzwerke verhalten sich ähnlich wie die Puls-Code-Modulation (PCM) in der digitalen Signalverarbeitung: Die Anzahl der Pulse pro Zeiteinheit, nicht die Amplitude, vermittelt die Grösse eines Signals. Das entsprechende mathematische Modell erfordert eine explizite Darstellung der Zeit und eignet sich daher am besten für die Verarbeitung von Zeitreihen.

Trotz des einfachen mathematischen Formalismus lassen sich viele Arten von neuronalen Netzen durch verschiedene Klassen von Gewichtsmatrizen realisieren. Um einige Beispiele zu nennen: Eine (Block)-Dreiecks-Gewichtungsmatrix stellt ein (geschichtetes) Feed-Forward-Netzwerk dar. Diagonale Blöcke stellen wiederkehrend verbundene Schichten (d. h. Gruppen von miteinander verbundenen Neuronen) dar, und alle Blöcke auf der zweiten oder höheren Nebendiagonalen stellen sogenannte Skip-Verbindungen dar. Nebendiagonale Blöcke in Toeplitz-Form sind Faltungsschichten (Convolutional Layer) – eine Struktur, die sich bei Bildverarbeitungsaufgaben als sehr nützlich erwiesen hat. Neben der Struktur der Gewichtsmatrix können wir auch wählen, welche numerische Art von Einträgen sie enthalten soll. Zum Beispiel können wir anstelle von reellen Zahlen ganzzahlige Gewichte verwenden, aus einer beliebigen Menge diskreter Gewichte auswählen, zum Beispiel binär (0, 1) oder ternär (–1, 0, 1), oder sogar eine komprimierte Kodierung der Gewichte verwenden. Zusammenfassend lässt sich sagen, dass es unzählige interessante Klassen von Gewichtungsmatrizen gibt, und jede von ihnen hat spezifische Auswirkungen auf die entsprechende Klasse von neuronalen Netzen und bietet spezifische Möglichkeiten zur Hardwarebeschleunigung.

Damit ein neuronales Netz überhaupt etwas Sinnvolles tun kann, müssen die freien Parameter, d. h. Gewichte und Bias-Terme, auf bestimmte aufgabenabhängige Werte gesetzt werden. Frühere technische

WAS IST NEUROMORPHE HARDWARE?

Der Begriff »neuromorph« ist offensichtlich ein Kunstwort aus *neuro-* und *-morph*, und er beschreibt Hardware, die in irgendeiner Weise von der Gestalt (Morphologie) biologischer neuronaler Systeme inspiriert ist. Da die Biologie viele Formen hervorbringen kann wird die Bezeichnung »neuromorphe Hardware« eher lose auf die Forschung angewandt, die das Ziel hat, (einige) neuronale Netzwerkmodelle in Hardware zu implementieren. Der beste Weg, dies zu erreichen, variiert je nach der Art des neuronalen Netzes, den Designbeschränkungen und den zu optimierenden Leistungskriterien, zum Beispiel Leistung, Latenz, Störungsrobustheit usw. Es gibt viele Freiheitsgrade im Designprozess, unter anderem:

- Welche Art von Netzwerken sollte verwendet werden, zum Beispiel Neuronen mit reelwertigen Zuständen, die sich kontinuierlich mit der Zeit ändern, oder mit diskreten Zuständen, die in diskreten Zeitschritten aktualisiert werden, oder spikenden Neuronen, die asynchron kommunizieren?
- Sollten diese Berechnungen durch analoge oder digitale Schaltungen realisiert werden?
- Sollte die Topologie der Netze auf eine bestimmte Struktur beschränkt werden, z. B. auf Feed-Forward-, rekurente oder Convolutional-Neural-Networks?
- Welche Komponenten des Netzes sollten direkt in Hardware implementiert werden, und welche sollten, wenn überhaupt, mit einem konventionelleren Prozessordesign berechnet werden?
- Kann das System in funktionale Module zerlegt werden, und wenn ja, wie sollten diese Module miteinander kommunizieren?

Jede Kombination dieser (und vieler weiterer) Designentscheidungen führt zu einer anderen Spezies neuromorpher Hardware-Designs. In der Praxis führt dies zu einem breiten Kontinuum von spezialisierten Multiprozessor-Designs auf der einen Seite bis hin zu vollständig analogen Umsetzungen biologischer neuronaler Netzwerkmodelle auf der anderen Seite, wo jedes Neuron und jede Synapse des neuronalen Netzwerkmodells ein dediziertes elektrisches Gegenstück hat. Neuromorphe Hardware umfasst daher mehrere verschiedene Technologien und nicht eine bestimmte, so dass eine genaue technische Definition des Begriffs schwierig ist. Stattdessen können wir einige charakteristische Merkmale oder Designprinzipien identifizieren, die (die meisten) neuromorphen Hardware-Designs von konventionelleren Ansätzen unterscheiden:

- hochparalleles Rechnen anstelle des sequentiellen Betriebs einer einzigen zentralen Verarbeitungseinheit (CPU),
- die Verwendung von verteiltem und dezentralisiertem Speicher anstelle eines zentralen Speichers und
- ein Systementwurf, der speziell für die Implementierung neuronaler Netze irgendeiner Form optimiert ist.

Wozu wird neuromorphe Hardware benötigt?

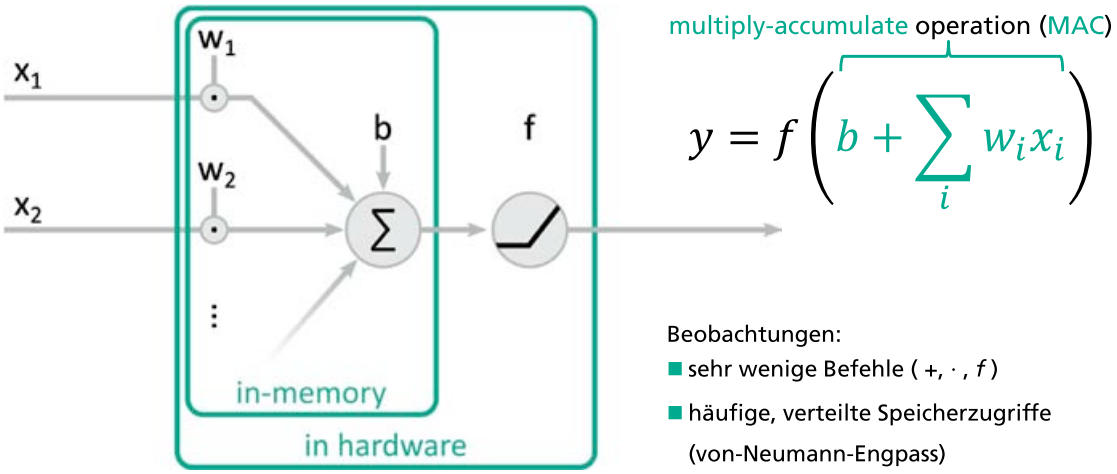

Bild 1: Ein einzelnes Neuron im neuronalen Netz gewichtet seine Eingangssignale (x_i) mit gelernten Gewichtswerten (w_i) und addiert sie auf. Diese Operation (MAC) kann in neuromorpher Hardware direkt durch analoge Ströme im Speicher realisiert werden.

Ansätze nahmen viele Merkmale tiefer neuronaler Netze vorweg, stützten sich aber stark auf Domänenwissen und Inspiration durch die in der Natur beobachteten Netzwerkstrukturen. Der wirkliche Durchbruch geschah Jahrzehnte später, als Verbesserungen in der Computertechnologie es plötzlich möglich machten, die Gewichtsmatrizen (und Bias-Terme) von hochstrukturierten Netzwerken (zum Beispiel tiefen Convolutional-Feed-Forward-Netzwerken) direkt zu optimieren oder zu trainieren, um Fehler bzw. Verluste bei extrem großen Datensätzen zu minimieren.

Da eine globale Optimierung solch großer nichtlinearer Systeme nahezu unmöglich ist, besteht der praktische Optimierungsansatz darin, einfache, erfolgsorientierte und gradientenbasierte Algorithmen zu verwenden, um die Parameter des Netzes schrittweise zu ändern, so dass der Fehler im Bezug auf einen Trainingsdatensatz minimiert wird.

Zwar ist dies für Spiking Neural Networks deutlich schwieriger (die zeitdiskrete Natur ihrer ereignisbasierten Kommunikation erschwert die Berechnung von Gradienten), aber es gibt Abhilfen, die es erlauben, ähnliche Werkzeuge auch für das Training von Spiking-Neural-Networks zu verwenden. Deshalb sind gradientenbasierte Methoden de facto zu einem so zentralen Bestandteil des Deep-Learnings geworden, dass *differential programming* sogar als genauere Bezeichnung für das gesamte Gebiet vorgeschlagen wurde.

Wie dieser kurze Überblick zeigt, kann man bei der Konstruktion neuronaler Netze an vielen Parametern drehen, und die möglichen Hardware-Implementierungen sind ähnlich reichlich vorhanden. Im Folgenden werfen wir einen kurzen Blick auf mehrere verschiedene Ansätze für den Entwurf neuromorpher Hardware, sortiert nach dem Grad, in dem sich das Netzwerkmodell direkt in der Hardware widerspiegelt.

Generische CoProzessoren und Grafikkarten

Da ein Großteil der Simulations- und Trainingszeit für neuronale Netze für MAC-Operationen aufgewendet wird, ist die Schlüsselinnovation bei den meisten Beschleunigerentwürfen eine effiziente Hardware-Implementierung der Matrix-Multiplikation. Die wohl flexibelste und allgemeinste Form von Hardware-Beschleunigern für neuronale Netze sind daher konventionelle Viel-Prozessor-Designs wie Grafikkarten (GPUs) oder andere »Number-Crunching«-Coprozessoren wie Tensor-Processing-/Streaming-Einheiten (TPUs), die für große und schnelle Matrix-Multiplikationen optimiert wurden. Sie gelten im Allgemeinen nicht als neuromorphe Hardware, aber die hohe Nachfrage nach Geräten für Deep-Learning-Anwendungen hat die Entwicklung einer neuen Generation von GPUs und TPUs vorangetrieben, die vollständig für generische parallele Rechenaufgaben optimiert sind. Software-Bibliotheken unterstützen dabei, immer mehr parallele Operationen weg von der CPU an solche CoProzessoren zu delegieren. Ihre Vielseitigkeit hat jedoch einen hohen Preis: Die Fähigkeit, beliebige Programme auszuführen, erfordert eine umfangreiche Steuerlogik, leistungsfähige arithmetische Logikeinheiten und ein Cache-, Speicher- und Bussystem, das für beliebigen Speicherzugriff und schnellen Datentransfer optimiert ist. Dieser Overhead ist für viele neuronale Netzwerkarchitekturen unnötig und erhöht Kosten und Stromaufnahme. Da die Hauptbeschleunigung solcher CoProzessoren durch die Beschleunigung der Matrix-Multiplikation erreicht wird, bringen sie für bestimmte Arten von Netzwerken wie SNNs, extrem spärlich verbundene Netzwerke oder Netzwerke mit nichtlinearen synaptischen Effekten kaum einen Nutzen.

Designs mit vielen Rechenkernen

Ähnlich, wenn auch stärker auf Anwendungen für neuronale Netze ausgerichtet, sind spezialisierte Vielkern-Designs, die die Aufgabe der Simulation oder des Trainings eines großen neuronalen Netzes auf viele unabhängige Prozessorkerne verteilen. Sie unterstützen typischerweise einen reduzierten Befehlssatz, der auf Anwendungen neuronaler Netze zugeschnitten und optimiert ist. Anstatt durch willkürlichen Zugriff auf den gemeinsamen Speicher implementieren diese Designs typischerweise ein effizientes Routing- oder Nachrichtenübergabesystem für den Informationsaustausch zwischen den Knoten. Trotz des Schwerpunkts auf An-

wendungen neuronaler Netze werden die Neuronen hier algorithmisch in Software emuliert, und die Daten fließen eher über ein gemeinsam genutztes Bussystem als über dedizierte Synapsen. Da diese Geräte keine der Komponenten eines neuronalen Netzes direkt in Hardware implementieren, sind sie keine echte neuromorphe Hardware im engeren Sinne, aber sie werden aufgrund ihrer nahezu identischen Anwendungsgebiete und Benutzerschnittstellen zusammen mit neuromorpher Hardware gefasst.

Digitale Deep-Learning-Beschleuniger

Es gibt eine weitere Klasse von digitalen Beschleunigern, die auf niedriger Ebene vollständig für die Implementierung von (bestimmten) tiefen neuronalen Netzen entworfen und optimiert sind. Hier wird der Betrieb einzelner Neuronen durch eine dedizierte digitale Logikschaltung nachgebildet, die die spezifischen MAC-Operationen realisiert, die für die relevante Netzwerkklasse erforderlich sind. Diese Bausteine eignen sich daher nicht für eine beliebige Codeausführung, sondern erfordern stattdessen die Bereitstellung der genauen Topologie und Koeffizienten des Netzwerks. Einmal konfiguriert, arbeiten sie als Black Box, die das bereitgestellte Netzwerk effizient ausführt und digitale Eingangssignale auf die digitalen Ausgänge des Netzwerks abbildet.

Solche Deep-Learning-Beschleuniger werden nicht allein, sondern als kleine Kerne verwendet, die in ein größeres Many-Core-System eingebettet sind. Ein solches modulares Design kann effektiver, flexibler und leichter skalierbar sein, insbesondere wenn die zu beschleunigende Netzwerkarchitektur spezifische Optimierungen zulässt. Beispielsweise verwenden Convolutional-Neural-Networks dieselbe Struktur synaptischer Gewichte wiederholt für verschiedene Neuronen, was sehr effizient durch die Wiederverwendung desselben Hardwaresubstrats in einem zeitmultiplexen Design implementiert werden kann, das das Netzwerk Neuron für Neuron aktualisiert. Im Allgemeinen sind in Feed-Forward-Netzwerken die Neuronen innerhalb einer Schicht aufgrund ihres Inputs bedingt unabhängig und können daher parallel verarbeitet werden. Hier kann eine effiziente Hardware-Lösung eine ganze Schicht auf einmal aktualisieren. Bei spärlich besetzten Gewichtsmatrizen kann eine optimierte Behandlung von Nullen die Leistung weiter verbessern, während Netzwerke mit niedrigpräzisen (zum Beispiel ternären) Gewichten durch wesentlich kompaktere Schaltungen implementiert werden können.

Analoge Deep-Learning-Beschleuniger

Wie bereits gezeigt, werden die mathematischen Modelle neuronaler Netze in der Regel in Form von reellen Zahlen angegeben. Anstatt sie also über diskrete digitale Schaltungen zu approximieren, besteht ein anderer naheliegender Ansatz darin, die reellwertigen Größen des Modells stattdessen durch reellwertige physikalische Größen wie analoge Spannungen, Ströme oder Ladungen darzustellen. Eine solche Verwendung des analogen Schaltungsentwurfs geht auf die frühesten Versuche des neuromorphen Hardware-Designs in den 1950er Jahren zurück, ist aber durch die digitale Revolution der Elektronik etwas aus der Gunst gefallen. Während die Anfälligkeit für Rauschen immer noch eine große Herausforderung für die meisten Anwendungen analoger Schaltungen darstellt, haben sich einige neuronale Netze als bemerkenswert robust gegenüber den Auswirkungen des Rauschens erwiesen.

Tatsächlich können einige Formen von Rauschen sogar dazu beitragen, die Robustheit des Systems zu verbessern! Andere Tücken des analogen Schaltungsentwurfs, zum Beispiel die Schwierigkeit, (Nicht-)Linearitäten im System präzise zu kontrollieren, sind für neuronale Netze viel weniger kritisch als für andere Anwendungen, da

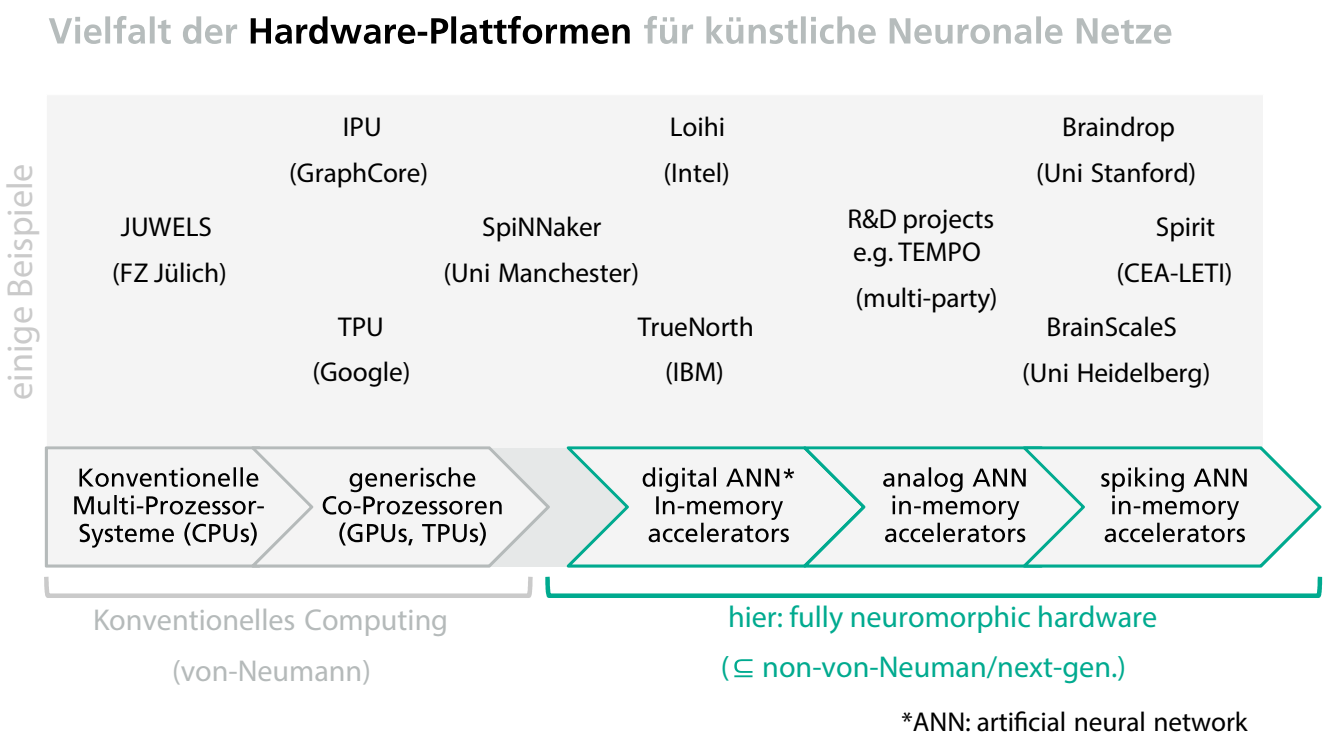

Bild 2: Die Bandbreite von Hardware-Plattformen, die zur Simulation neuronaler Netze eingesetzt werden, erstreckt sich von klassischen Multi-Core-Systemen und Großrechnern über spezialisierte Prozessoren bis hin zu anwendungsspezifischen Schaltungen in digitaler, analoger oder asynchron ereignisgesteuerter Hardware.

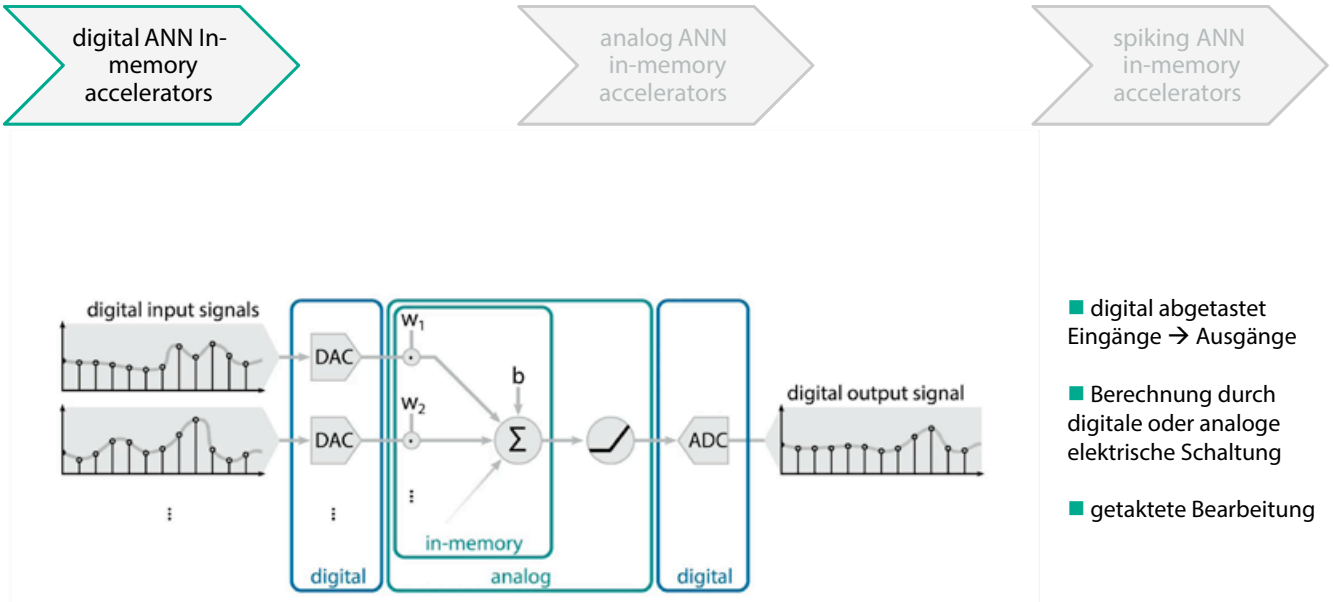

Bild 3: Um auf einer digitalen Plattform eine effiziente Berechnung der Neuronen durch analoge Operationen direkt im Speicher realisieren zu können, muss ein digitales Eingangssignal in analoge Ströme umgewandelt, verarbeitet und das Ergebnis wieder digitalisiert werden.

die Netze über genügend Freiheitsgrade verfügen, um solchen Defekten entgegenzuwirken (vorausgesetzt natürlich, dass die Defekte bekannt sind). Trotz dieser Herausforderungen und Einschränkungen, die letztlich den Übergang zum digitalen Schaltungsentwurf verursacht haben, hat die Implementierung neuronaler Netze im analogen Bereich große Vorteile. Erstens ist das kontinuierliche Modell eines Neurons dem eines Logikgatters bemerkenswert ähnlich – mit einem entscheidenden Unterschied: Neuronen können hinsichtlich ihrer Parameter abgeleitet werden, was für Deep Learning entscheidend ist. Deep Learning bietet daher einen Rahmen, um analoge Schaltungen in einer Weise zu optimieren, die nicht direkt auf digitale Schaltungen angewendet werden kann. Zwar ist es möglich, das Verhalten kontinuierlicher Neuronenmodelle durch digitale Schaltungen zu approximieren, aber dies kann zu einer hohen Zahl von Transistor- und Logikgattern führen, von denen jedes für sich eine Komplexität aufweist, die der einer analogen Implementierung des Neuronenmodells gleichkommt.

Im analogen Ansatz wird die Multiplikation und Addition stattdessen durch direkte Anwendung der Ohmschen und Kirchhoffschen Gesetze realisiert, d. h. durch die Wahl geeigneter Widerstände zur Darstellung der einzelnen synaptischen Gewichte und die Akkumulation der resultierenden Ströme. Dies ermöglicht auch Anwendungen mit ultraniedriger Leistung, möglicherweise zuz Lasten der Rauschunterdrückung, und vermindert die Notwendigkeit, auf das Einschwingen der Signale zu warten, was asynchrone Designs mit niedriger Latenz ermöglicht.

Aufgrund starker Markteintrittsbarrieren, zum Beispiel hohe Herstellungskosten und lange Entwicklungszyklen, hat bisher nur eine vergleichsweise geringe Anzahl analoger Deep-Learning-Beschleuniger das Licht der Welt erblickt. Es ist jedoch bereits sehr viel Literatur darüber geschrieben worden, und wenn die Zahl der jüngsten Neugründungen und Forschungsprojekte auf diesem Gebiet ein Hinweis darauf ist, dann gibt es auch ein erhebliches und wachsendes kommerzielles und akademisches Interesse.

Spiking-Neural-Network-Beschleuniger

Spiking-Neural-Networks (SNNs) sind derzeit ein Außenseiter in der Welt des maschinellen Lernens, aber passende neuromorphe Hardware wird ein weiterer wichtiger Zweig der hardwarebeschleunigten, eingebetteten KI werden. Im Gegensatz zu herkömmlichen neuronalen Netzen widersetzt sich der rein ereignisgesteuerte Betrieb von Spiking-Neural-Networks dem einfachen mathematischen Rahmen der kontinuierlichen Funktionsberechnung und der periodischen Abtastung. Eine effiziente Implementierung solcher Netzwerke ist daher sowohl für getaktete digitale Logik als auch mit konventioneller Programmierung schwierig. In Kombination mit der erhöhten Komplexität der Trainingsalgorithmen für SNNs könnte dies die relativ geringe Popularität dieser Netzwerke erklären.

Allerdings sind die Eigenschaften von »spikenden« Neuronen, die als Haupthindernisse für effiziente Software-Implementierungen erscheinen (zum Beispiel zeitliche Integration und Tiefpassfilterung von Signalen und eine durch Anstiegsflanken getriggerte Erzeugung von Impulsen), in der Signalverarbeitung alltäglich und können durch einfache analoge Schaltungen implementiert werden. Infolgedessen ist die Zunahme der Komplexität beim Wechsel von einem analogen zu einem spike-basierten Netzwerkdesign gering. Im Gegenteil, da der spike-basierte Ausgang jedes Neurons ein binäres Signal ist, das allein durch Tiefpassfilterung in ein analoges Signal umgewandelt werden kann, kann spike-basierte neuromorphe Hardware das Beste aus beiden Welten kombinieren: die äußerst energieeffiziente Berechnung analoger Schaltungen und die binäre Übertragung von Signalen über Spikes, was die Rauschanfälligkeit verringert und das Routing und Puffern vereinfacht.

Genau wie bei digitalen Hardware-Beschleunigern kann die Kommunikation zwischen einzelnen Neuronen eines SNNs daher entweder über dedizierte elektrische Leitungen oder über ein (digitales) Paket-Routing-System implementiert werden,

wobei das populärste Protokoll »address event representation« ist, bei der jeder Puls als Paket mit der »Adresse« des Neurons, von dem er stammt, übertragen wird. Während ein solches Routing-System die Skalierbarkeit des Systems durch Zeitmultiplexing derselben Kommunikationskanäle erheblich verbessert, erfordert es eine ausgeklügelte Planung und geringe Latenzen, die mit zunehmender Anzahl miteinander verbundener Neuronen unerschwinglich werden können. Ein hybrider Ansatz, der viele Kerne mit vollständiger interner Konnektivität durch dedizierte Leitungen verwendet, die über ein gemeinsames Bussystem miteinander verbunden sind, ist daher ein beliebter Kompromiss.

Warum ist neuromorphe Hardware plötzlich wichtig?

Nach diesem kurzen Überblick könnte man sich fragen, warum neuromorphe Hardware so plötzlich zu einem heißen Thema unter KI-Forschern und ASIC-Entwicklern geworden ist. Keine dieser Ideen scheint neu genug zu sein, um diesen Popularitätsanstieg zu rechtfertigen. Tatsächlich wurden seit den Anfängen der Forschung im Bereich der künstlichen Intelligenz und der Informatik in den 1950er Jahren immer wieder ähnliche Ideen vorgeschlagen. Warum sollten wir also heute in neuromorphe Hardware investieren, und warum ist dies nicht schon früher geschehen? Das naheliegendste Argument ist rein opportunistischer Natur: Nie zuvor hatten neuronale Netze eine ausreichende Größe, um für komplexe, datengesteuerte Anwendungen praktisch nutzbar zu sein. Jetzt, mit den Durchbrüchen bei Bildklassifikationswettbewerben in den letzten Jahren, haben sich neuronale Netze endlich für kommerzielle Anwendungen bewährt und sind seitdem in der Öffentlichkeit und in der Industrie massiv in Erscheinung getreten. Diese wachsende Popularität hat entsprechend zu einem erhöhten Bedarf an effizienter Hardware geführt, auf der neuronale Netze betrieben werden können.

Die Anwendungen in Bereichen wie Bildverarbeitung, Computerspielen, Textanalyse, Audioverarbeitung und Datenwissenschaften, zum Beispiel in der medizinischen Bildanalyse, haben sich diversifiziert und sind komplexer geworden, wobei die Anzahl der gelernten Gewichtskoeffizienten von Hunderttausenden bis zu einer schwindelerregenden Milliarde reichen können.

Die Bandbreite der Anwendungen wird voraussichtlich noch zunehmen, da eine wachsende Zahl mobiler Geräte vom Smartphone bis hin zu autonomen Fahrzeugen neuronale Netze für anspruchsvolle Bilderkennungsaufgaben bereits nutzen (oder in Kürze nutzen werden) und entsprechende Rechenleistung benötigen.

Bislang reiten wir auf der Welle immer besserer CPUs und GPUs, und allein der technologische Fortschritt könnte die wachsende Nachfrage aufrechterhalten, aber da wir das Ende des Moore'schen Gesetzes zu erleben beginnen, brauchen wir grundlegend neue Ideen. Die derzeitige hochmoderne 7-nm-CMOS-Technologie stößt an physikalische Grenzen, und es scheint unwahrscheinlich, dass wir die Größe, den Stromverbrauch oder die Latenz noch viel weiter verringern können. Gleichzeitig sind die Zeit für das Training, der Stromverbrauch und die Einstandskosten für Systeme, die in der Lage sind, größere neuronale Netze nach dem neuesten Stand der Technik zu simulieren, in die Höhe geschnellt, wobei das begrenzte Leistungsbudget mobiler Geräte ein limitierender Faktor für viele potenziell interessante Anwendungen war.

Silizium-Technologie treibt Neuro-Hardware voran

Natürlich hat auch die neuromorphe Hardware von den technologischen Fortschritten in der Elektronikfertigung in den letzten Jahrzehnten profitiert. Neben einem neuen Markt ist daher die Verfügbarkeit neuer Technologien ein weiterer Grund für ein erneutes Interesse an der Entwicklung neuromorpher Hardware. FinFets, FD-SOI und floating Multi-Gate-Mosfet-Transistoren sowie spezielle Neurotransistoren (vMOS) haben extrem stromsparende und neuartige neuromorphe Hardware-Designs ermöglicht.

Da neuronale Netze einen beträchtlichen Speicherbedarf für die Speicherung der Netzwerktopologie und der synaptischen Gewichte haben, kann neuromorphe Hardware auch viel von neuen Trends in der Speichertechnologie profitieren. Mit kleinen Feature-Größen von 28 nm und darunter

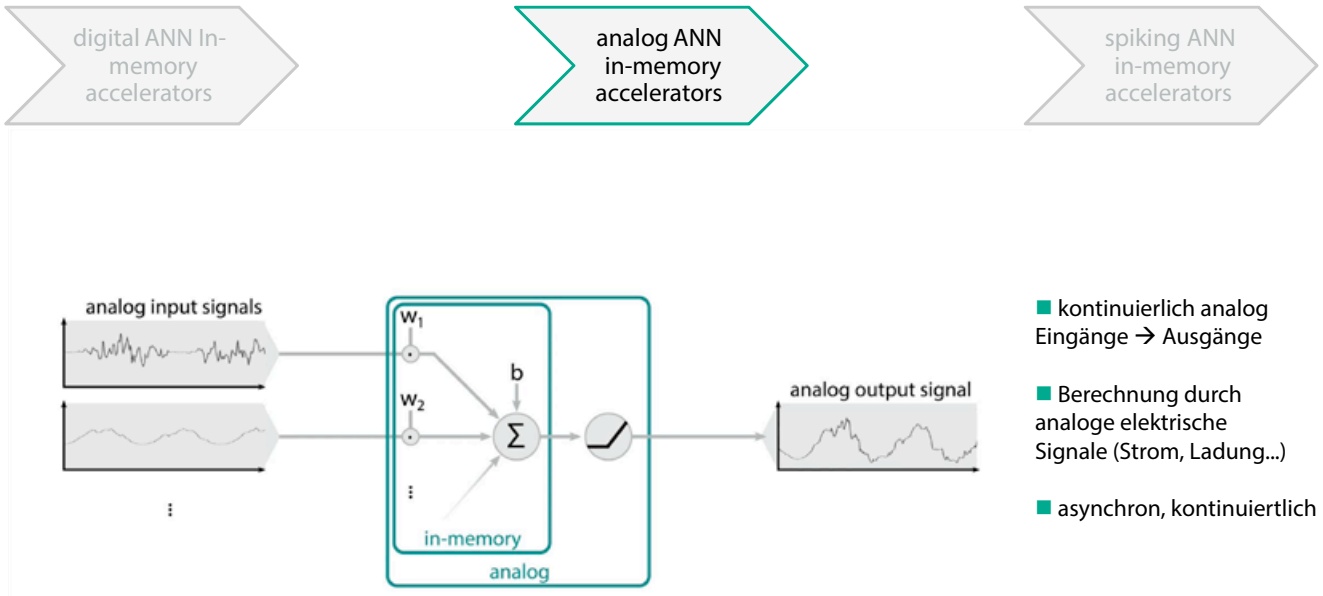

Bild 4: In analoger neuromorpher Hardware werden die Ein- und Ausganssignale der Neuronen als analoge Ströme übertragen, und die gesamte Verarbeitung kann in einer ungetakteten analogen Schaltung realisiert werden.

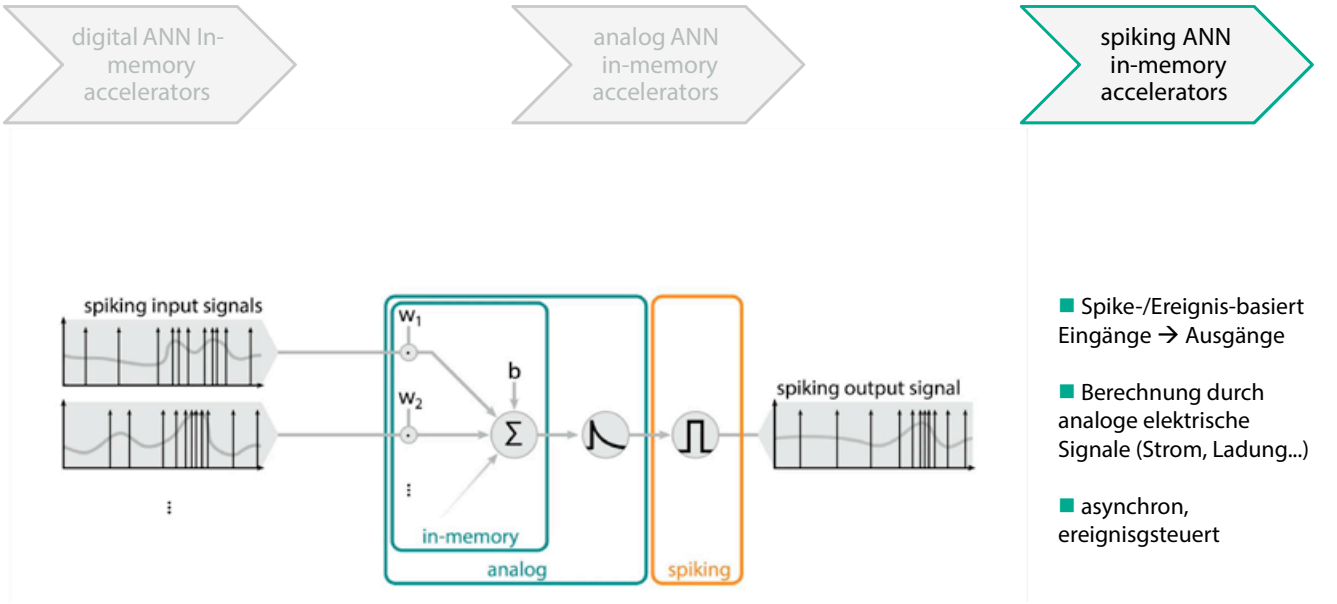

Bild 5: In spike-basierter neuromorpher Hardware werden die Ein- und Ausgangssignale der Neuronen durch Pulsfolgen kodiert, die als binäre Signale übertragen, aber wie analoge Signale verarbeitet werden können.

und Fortschritten bei dynamischem DRAM und statischem SRAM ist es möglich geworden, Netzwerke von praxisrelevanter Größe direkt im Silizium zu speichern und Daten direkt dort zu verarbeiten, wo sie gespeichert sind. Während dieses neue Paradigma des In-Memory-Computing keineswegs auf neuromorphe Hardware beschränkt ist, kann die hochgradig verteilte Struktur neuronaler Netze diesen Vorteil besonders gut nutzen und so den Speicherengpass überwinden, unter dem herkömmliche von-Neumann-Architekturen leiden.

Da sich die Netzwerkkoeffizienten eines trainierten Neuronalen Netzes während der Inferenz (typischerweise) überhaupt nicht mehr ändern, sind neu aufkommende nichtflüchtige Speichertechnologien (eNVM,) für neuromorphe Hardware besonders interessant. Konkurrierende Technologien wie Charge-Trap-Flash-Speicher (CTF), ferroelektrische Feldeffekttransistoren (FeFets), resistive RAM (ReRAM), leitende Brücken-RAM (CBRAM) und Phasenwechselspeicher (PCM) nutzen alle verschiedene physikalische Phänomene aus, um eine nichtflüchtige Speicherung auf dem Chip zu ermöglichen, wobei viele von ihnen die Speicherung von Analogwerten mit einer Multi-Bit-Auflösung unterstützen, was für analoge Hardware-Beschleuniger kritisch ist und den für den Speicher benötigten Platz reduziert.

LITERATUR

[1] Die Englische Originalversion dieses Artikels mit allen wissenschaftlichen Quellenangaben finden Sie im Konferenzband der embedded world Conference 2020, erhältlich auf shop.weka-fachmedien.de:

[2] Sikora, A. (Hrsg.): embedded world Conference 2020 – Proceedings. Hierin: Leugering. J.: A visit to the neuromorphic zoo, p. xx-yy. WEKA Fachmedien, Haar, 2020.

Wie entwickelt sich neuromorphe Hardware weiter?

Natürlich ist es schwierig, die Zukunft vorherzusagen, insbesondere für ein Gebiet, das alle »Jahreszeiten«, einschließlich des gefürchteten »KI-Winters« bereits mehrmals durchlaufen hat. Aber mit neuen Technologien am Horizont, wie zum Beispiel 3D- und Wafer-Scale-Integration, Nanodraht-Transistoren, Kohlenstoff-Nanoröhren auf dem Chip, Silizium-Photonik, Spintronik, immer kleineren mikro- und nanoelektromechanische Systeme (MEMS, NEMS), integrierten Sensor-Prozessor-Systemen und mehr, ist es schwer, nicht optimistisch an die Zukunft neuromorpher Hardware zu denken.

Der technologische Fortschritt wird die neuromorphe Hardware voraussichtlich in neue Anwendungsbereiche bringen, wo sie die Energieaufnahme, die Latenzzeit oder die Kosten bestehender Lösungen reduzieren kann. Beispielsweise werden CoProzessoren für KI bereits in moderne Smartphones eingebaut, um die CPU-Last bei KI-Anwendungen zu reduzieren und damit die Akkulaufzeit zu verlängern. Am anderen Ende des Spektrums zeigt die steigende Nachfrage nach Hochleistungs-Computing-Clustern und Cloud-Diensten, die »Deep-Learning-as-a-Service« bieten, einen Markt für serverseitige neuromorphe Hardware-Coprozessoren.

Neuromorphe Hardware kann auch Anwendungen ermöglichen, die derzeit prinzipiell möglich, aber noch nicht wirtschaftlich tragfähig sind, wie zum Beispiel natürlichsprachliche oder gestengestützte Benutzerschnittstellen zur Steuerung verschiedenster elektrischer Geräte, zum Beispiel im Kontext des Internet der Dinge oder im Haushalt. Die Industrie könnte neuromorphe Hardware einsetzen, um selbst Low-Level-Prozesse in der Fertigung anpassungsfähiger oder reaktionsfähiger zu machen oder die Interaktion zwischen Mensch und Maschine zu verbessern.

Schließlich könnte der Einsatz neuromorpher Hardware sogar Lösungen für das maschinelle Lernen ermöglichen, die derzeit gänzlich unmöglich sind, wie zum Beispiel anspruchsvolle Echtzeit-Verarbeitung und Fusion komplexer, hochdimensionaler Sensordaten und die intelligente Echtzeitsteuerung anspruchsvoller Roboter oder Produktionsanlagen. Energieeinsparungen könnten neuartige mobile Anwendungen wie groß angelegte verteilte Sensornetzwerke oder autonome Systeme ermöglichen, die intelligent genug sind, um eigenständig zu agieren, und die robust genug sind, um in schwierigen Umgebungen zu bestehen. Durch die Optimierung der Siliziumfläche könnte neuromorphe Hardware auch den Weg in miniaturisierte Sensoren finden, zum Beispiel in einnehmbare medizinische Sensoren. (jk)