micromachines

MDPI

Review

Advancements in Microprocessor Architecture for Ubiquitous AI—An Overview on History, Evolution, and Upcoming Challenges in AI Implementation

Fatima Hameed Khan, Muhammad Adeel Pasha * and Shahid Masud *

Department of Electrical Engineering, Lahore University of Management Sciences (LUMS), Lahore, Punjab 54792, Pakistan; 20060029@lums.edu.pk
* Correspondence: adeel.pasha@lums.edu.pk (M.A.P.); smasud@lums.edu.pk (S.M.)

Abstract: Artificial intelligence (AI) has successfully made its way into contemporary industrial sectors such as automobiles, defense, industrial automation 4.0, healthcare technologies, agriculture, and many other domains because of its ability to act autonomously without continuous human interventions. However, this capability requires processing huge amounts of learning data to extract useful information in real time. The buzz around AI is not new, as this term has been widely known for the past half century. In the 1960s, scientists began to think about machines acting more like humans, which resulted in the development of the first natural language processing computers. It laid the foundation of AI, but there were only a handful of applications until the 1990s due to limitations in processing speed, memory, and computational power available. Since the 1990s, advancements in computer architecture and memory organization have enabled microprocessors to deliver much higher performance. Simultaneously, improvements in the understanding and mathematical representation of AI gave birth to its subset, referred to as machine learning (ML). ML includes different algorithms for independent learning, and the most promising ones are based on brain-inspired techniques classified as artificial neural networks (ANNs). ANNs have subsequently evolved to have deeper and larger structures and are often characterized as deep neural networks (DNN) and convolution neural networks (CNN). In tandem with the emergence of multicore processors, ML techniques started to be embedded in a range of scenarios and applications. Recently, application-specific instruction-set architecture for AI applications has also been supported in different microprocessors. Thus, continuous improvement in microprocessor capabilities has reached a stage where it is now possible to implement complex real-time intelligent applications like computer vision, object identification, speech recognition, data security, spectrum sensing, etc. This paper presents an overview on the evolution of AI and how the increasing capabilities of microprocessors have fueled the adoption of AI in a plethora of application domains. The paper also discusses the upcoming trends in microprocessor architectures and how they will further propel the assimilation of AI in our daily lives.

Keywords: artificial intelligence; microprocessors; instruction set architecture; application-specific integrated circuits; real-time processing; machine learning; intelligent systems; automation; multicores

Citation: Khan, F.H.; Pasha, M.A.; Masud, S. Advancements in Microprocessor Architecture for Ubiquitous AI—An Overview on History, Evolution, and Upcoming Challenges in AI Implementation. *Micromachines* **2021**, *12*, 665. https://doi.org/10.3390/mi12060665

Academic Editor: Piero Malcovati

Received: 5 May 2021
Accepted: 3 June 2021
Published: 6 June 2021

1. Introduction

Artificial intelligence (AI) is a thriving tool that has coupled human intelligence and machine efficiency to excel in various disciplines of life. The idea of building intelligent machines is even older than the field of AI itself. In 1950, Alan Turing presented the possibility of implementing an intelligent machine and gave the parameters to judge its intelligence, known as the Turing test [1]. The term "artificial intelligence" was first mentioned in the Dartmouth Conference in 1956, which was attended by those who later became the leading figures of this field. Early AI research included several programs and methodologies, such as General Problem Solver [2], Theorem Prover [3], natural

language processor ELIZA [4] and the discovery of the perceptron [5]. Intrigued by the success of these projects, the Defense Advanced Research Projects Agency (DARPA) also invested massively in this field to utilize machine intelligence for various national security projects [6]. After this first wave, AI encountered technological barriers that curtailed the vast expectations of AI. In the following decades, as the market for personal computers (PCs) was established, the paradigm of AI shifted to a knowledge-based-system, also known as Expert Systems [7]. It was programmed using symbolic programming languages such as LISP or Prolog. Expert Systems were used to implement human expert knowledge in a machine that makes decisions based on stored information [8]. Due to problems in accuracy and efficiency, Expert Systems could not find widespread acceptability in the market.

Finally, the increase in computing power and development of more sophisticated mathematical modeling tools gave birth to a new AI paradigm that became more successful than before. The more advanced subset of AI algorithms in the form of machine learning (ML) addressed the complex problems of AI and showed a promising way forward because of its ability to make autonomous decisions based on previous learning and the scenarios at hand [9]. The algorithms used in ML can find applications in diverse use cases, such as the use of decision trees to monitor the depth of anesthesia [10] and support vector machines (SVM) in financial research [11]. Artificial neural networks (ANN) have not only outperformed other ML algorithms but also surpassed human intelligence in specific tasks, e.g., image classification on an ImageNet dataset [12]. It motivated researchers to delve deeper into the ANN structure, which resulted in networks with more parameters, layers, and operations, which came to be classified as deep neural networks (DNN) [13]. DNNs can be further divided into structured network techniques for various applications, i.e., recurrent neural networks (RNN) and transformer networks that mainly focus on natural language processing. In recent years, AI has revolutionized society by covering a wide range of applications, from the simplest smartphones in our hands to security and surveillance [14] and autonomous vehicle controls [15]. Currently, it is being used in agriculture [16], healthcare departments for diagnosing different diseases [17,18], business and finance [19,20], robotics [21], web searches [22], computer vision [23,24], etc. A taxonomy of AI and its various sub-fields is shown in Figure 1.

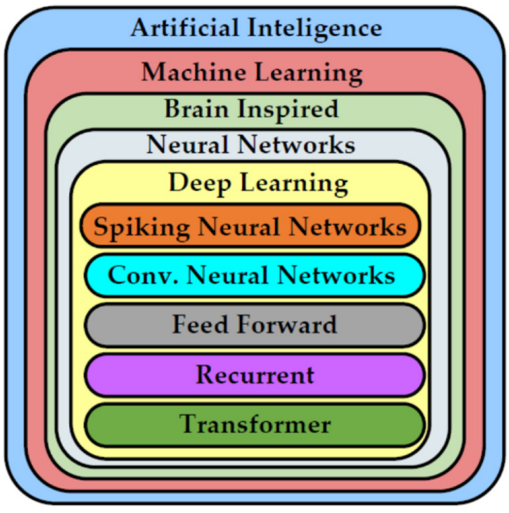

Figure 1. Taxonomy of AI and its sub-fields.

Initially, limited computing power and memory restricted the growth of AI. Scientists could not go beyond certain performance limits of computers, as the execution time increased exponentially with the algorithmic complexity. Quillian [25] used a vocabulary of only 20 words to present his work on natural language due to computer memory constraints. Even today, when processors are able to deliver 221 K MIPS at 5.0 GHz

(Intel core i7-8086 K [26]) compared to the 80 to 130 MIPS in the fastest supercomputer from 1976 (Cray-1 [27]), the architecture for AI applications still demands high resource management to fulfill its real-time requirements. The remarkable research potential of ANN created an appetite for the drastic increase in computational and memory resources as the number of layers in DNNs exceeded 10,000 [28]. Billions of multiply-and-accumulate (MAC) operations necessitated the development of a suitable hardware platform for DNN implementation. It is also worth mentioning that each MAC operation requires frequent memory accesses: three memory reads and one memory write per MAC in worst case, i.e., with no data reuse [9]. Apart from computation and memory requirements, latency is also one of the vital constraints for several real-time applications like self-driving cars, where the slightest processing delay can result in catastrophic consequences [9].

Along with the advancements in intelligent algorithms, the adoption of AI is also driven by the improvement in performance of microprocessors that carry out all the required operations. In 1971, Intel launched the Intel 4004 chip along with its chipset, and the first commercial microprocessor came into being. Intel 4004 [29] was a 4-bit device made up of 2300 transistors and would execute 92600 instructions per second (IPS) at a clock frequency of 740 kHz. Subsequently, a diverse range of microprocessors was developed and the market for computers became ubiquitous. The IBM PC, based on an 8-bit microprocessor (Intel 8088, Intel Corporation, Santa Clara, CA, USA), was introduced in 1979 [30] and became a standard for personal computers (PCs).

Researchers attempted to further increase the processing bandwidth, but the complex micro-architecture became a big hurdle in terms of instruction-set decoding. It led to the idea of reduced instruction set computer (RISC) [31] architecture, which was demonstrated by Acorn RISC Machines (also known as ARM, ARM Ltd., Cambridge, England) based on a 32-bit RISC processor [32]. Driven by Moore's Law [33], the exponential reduction in transistor size over the years paved the way for 4-bit microprocessors to expand to 64-bit bandwidth while introducing many complex architectural optimizations, i.e., pipelining, cache memory, virtual memory, and application-specific very large instruction words (VLIW) [34]. Major limitations to the implementation of AI were addressed by advancements in different functional modules of microprocessors. For example, an increase in memory bandwidth reduces the operation of frequent data swapping from the memory, which is a major performance bottleneck of AI algorithms, especially for DNNs [35]. AI processors accomplished higher amounts of computing power and reduced memory access time by overcoming the famous power wall [36] and memory wall [37] issues. Still, the general-purpose computers with high-end microprocessors were not able to satisfy the ever-increasing demands of ML and DNN algorithms because of limited number of cores and their monolithic architecture.

Later, graphic processing units (GPUs) equipped with high performance parallelism and memory bandwidth were widely adopted for implementing power-hungry DNN algorithms; this is currently a popular approach for AI implementation [38]. However, in recent years, research in AI architecture has moved towards AI or NN application-specific hardware (commonly known as AI accelerators), which has shown promising results [39]. Many AI-specific accelerators, implemented on FPGA or ASIC, have been proposed to improve energy efficiency, such as DaDianNao [40], PuDianNao [41], Cambricon [42], etc. An innovative industrial approach was developed by Google, called the tensor processing unit (TPU), [43] which reduces the computational precision to accelerate AI operations for the TensorFlow software. Many efficient solutions are emerging for smart embedded processors, but there is room for innovation in this emerging area.

This interaction between the evolution of AI and advancement in microprocessors is following a symbiotic trend that motivated us to write this review paper. Although, many survey papers have been published that give an overview of microprocessor development [44,45] and the emergence of AI [6,9,46] separately, this paper is intended to give an overview of how the progress in microprocessors has enabled the implementation of AI in a broad spectrum of real-life applications. We have restricted this work to the im-

plementation of general-purpose architecture for AI applications because domain-specific hardware is yet an emerging branch of AI [39]. More time needs to pass to evaluate the benefits and directions that AI accelerators will take.

The rest of the of the paper is organized as follows: Section 2 discusses the history and evolution of AI before the exponential improvement in microprocessor capabilities. In Section 3, trends in microprocessors that eventually resulted in the latest accolades in the AI field are elaborated. Section 4 includes a discussion on the confluence of AI and microprocessors and the impact each of these had on the other over the course of the last few decades. Section 5 includes a brief overview of the roadmap followed by trends and the challenges that are expected in future. Section 6 concludes the paper with important observations.

2. Origin and Evolution of AI

The idea of making a machine that autonomously performs different tasks has fascinated the human mind even before the advent of digital computers. In 1930, Vannevar Bush presented a set of rules to automatically solve differential equations [47]. After a few years, the British mathematician Alan Turing established the concept of solving any algorithmic problem on a machine that is popularly known as the Turing machine [48]. Practical implementation of AI was made possible by the invention of programmable computers in 1944. Turing's famous work on measuring machine intelligence gave a jump start to AI. The Turing test [1] is based on the idea of distinguishing between a computer and a human being by observing the conversations between them. In 1952, the first AI program for checkers demonstrated the learning ability of computers [49], and later it was significantly improved by Samuel in 1959 [50] to challenge any decent player of that time. Then, the logic theorist approach was developed using critical concepts of AI to prove some of the geometric theorems [51].

The field of AI officially began with a conference at Dartmouth College in 1956, where the term "artificial intelligence" was first coined by John McCarthy. Many researchers, including the creators of the logic theory machine (Newell and Simon), attended this session and everyone was very curious and positive about the success of AI in the future. The same duo, Newell and Simon, built another AI machine, the General Problem Solver (GPS) [2], and claimed that the GPS could solve any problem given a well-formed description. The GPS, however, did not meet expectations when it came to solving complex problems requiring run-time information handling. It was not that the machine could not solve those problems, but that the time taken to compute answers for complex problems was so long that a machine was impractical. In 1958, McCarthy introduced the first AI programming language, list processing language (LISP) [52] which empowered scientists to store information in the form of objects instead of numbers. Meanwhile, Rosenbaltt's discovery of the basic unit of neural networks, the perceptron algorithm, gave birth to the concept of connectionism. It was predicted to be the "embryo of an electronic computer," but further research into this problem was halted due to the work of Minsky and Seymour in 1969 [53] that highlighted severe limitations of the perceptron and Rosenbaltt's prediction algorithm.

Rapid growth in the research of AI led to the creation of the first ever industrial robot, Unimate, in 1961, to work on assembly lines [54]. Furthermore, the first ever LISP program, known as symbolic automatic integrator (SAINT), to heuristically solve calculus problems was developed by James Slagle [55]. Another approach for mathematical computation called STUDENT was presented by Daniel Bobro [56], which could solve algebra word problems. It is cited as one of the earliest natural language processors, as it was programmed to accept natural language as its input. Later, Joseph Weizenbaum made the first interactive computer program, ELIZA [4], which could hold conversations with human subjects based on some grammatical rules. Weizenbaum was himself surprised to see that many people failed to differentiate between ELIZA and humans. Then, Stanford University produced the first general purpose robot, Shakey [57], which was capable of reasoning on its actions and combined logical reasoning with physical response. The fusion

of the two fields, computer vision and natural language processing (NLP), added a new flavor to the field of AI. In a similar vein, another computer program, SHRDLU, was designed to have conversations in English, to plan robot operations, and to apply different actions on simple toy objects [58].

Despite all this success in developing AI algorithms and applications, the most advanced systems were able to handle only a limited scope of problems. In the beginning, the field of AI was perceived with great optimism in all areas of applications. When AI failed to meet expectations, in the 1970s there was a drastic loss of enthusiasm for AI-related research activities both in academia and industry. The main impediment was that AI was unable to overcome the computational barriers in real-time implementation due to the unavailability of powerful processing devices. Scientists had realized that the exponential growth of problem complexity prevented the execution of computer programs in real time. The initial hype of AI attracted many agencies like DARPA, the National Research Council (NRC), and other governmental organizations globally to invest huge funds in this field for different objectives.

Two famous reports, ALPAC (from the US) [59] and Lighthill (from the UK) [60], showed great disappointment in yielding little from AI technology with large investments. Another reason for the drop in AI was the book by Minsky and Seymour, titled "Perceptrons," in which the authors argued against a learning machine and that only fixed networks were possible. This book had a widespread impact and stopped any further progress in connectionism for almost a decade. Historians called this period an "AI winter".

In the 1980s, the rise of Expert Systems gave new life to AI. The working of Expert Systems was based on a knowledge database with the rules and facts of a particular domain and an inference engine to manipulate the stored symbols [7]. This approach was used to gain human expertise in that specific application. Though Expert Systems became popular in the 1980s, its development had begun in 1965 by Edward Feigenbaum. He built Dendral, which was an Expert System specialized in identifying organic compounds [61]. With passing time, Expert Systems gained commercial attention and MYCIN was built to diagnose blood infectious diseases and prescribe a suitable medicine for them [8]. Another AI programming language, Prolog, was also created in 1972, and mainly focused on linguistic processing [62]. The development of a program, XCON was marked as an enormous success for Expert Systems. It was created for the Digital Equipment Corporation (DEC) to automatically choose computer components according to the requirements. XCON enabled the company to save USD 40 million per year in 1986 [63]. It was the time when Expert Systems were considered to have rejuvenated the field of AI, as they were oriented to solving practical problems at the industrial level. In the late 1980s, many companies were aiming to develop or maintain Expert Systems [64]. Figure 2 shows the generic workflow of an Expert System.

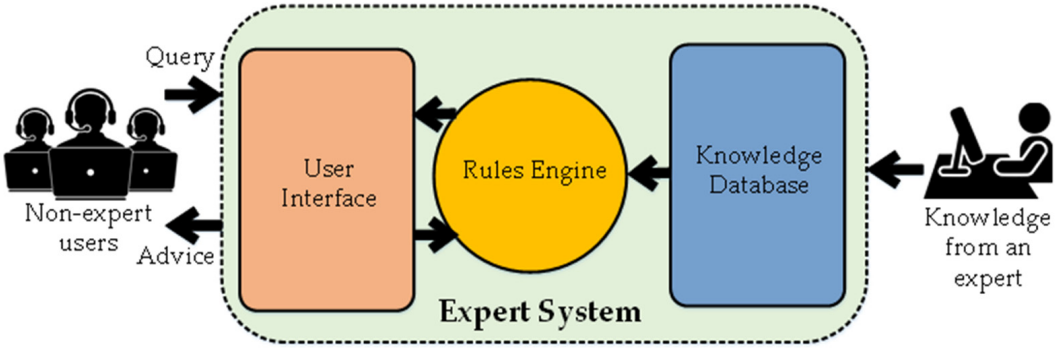

Figure 2. Generic operation of an Expert System.

This surge in the field of AI pulled many investments, especially in Japan, where the government dedicated a huge amount of funds to their Fifth Generation Computer System project [65]. Its purpose was to build a machine that could communicate, translate lan-

guages, recognize pictures, and argue like a human. This decade also witnessed the revival of connectionism, as John Hopfield proposed a recurrent neural network, Hopfield Net, in 1982 [66]. This neural network offered a remarkable improvement in learning process, which proved to be a hard competitor for the symbolic and logical AI. Rehmalhart et al. [67] also contributed to the revival of neural networks (NNs) by suggesting a backpropagation technique to train NN models. However, it was still too early to implement useful NN applications, as the lack of training data and restricted computing resources ultimately set the limit on its growth.

At the start of the 1990s, Expert Systems also lost steam because of the difficulty in knowledge acquisition and its analysis in real time. The knowledge acquisition process discovered the rules of the real word and linked them to solve any problem in a human manner, but it was not possible to reflect all human skills and feed them into Expert Systems [64]. Moreover, Expert Systems did not have the capability to learn, adapt, and evolve based on their interaction with the users. The situation was worsened by the AI symbolic languages LISP and Prolog, as they led to integration issues in complex systems. The environment of Expert Systems was not compatible with applications programmed in other languages (e.g., C). The rising performance of PCs led to the fading out of Expert Systems, which resulted in a loss of millions of USD for the industries based on Expert Systems [68]. After this downfall, researchers hesitated to put efforts in this field of AI, but many continued to work under different banners such as "machine learning," "intelligent systems," and "knowledge-based systems." The re-branding of all these terms made the survival of AI possible in the future. It also gave fine boundaries to the sub-branches of AI that further evolved with advancements in AI. Figure 3 shows the evolution of AI, going from a simple program solver to deep learning by addressing more and more complex application domains.

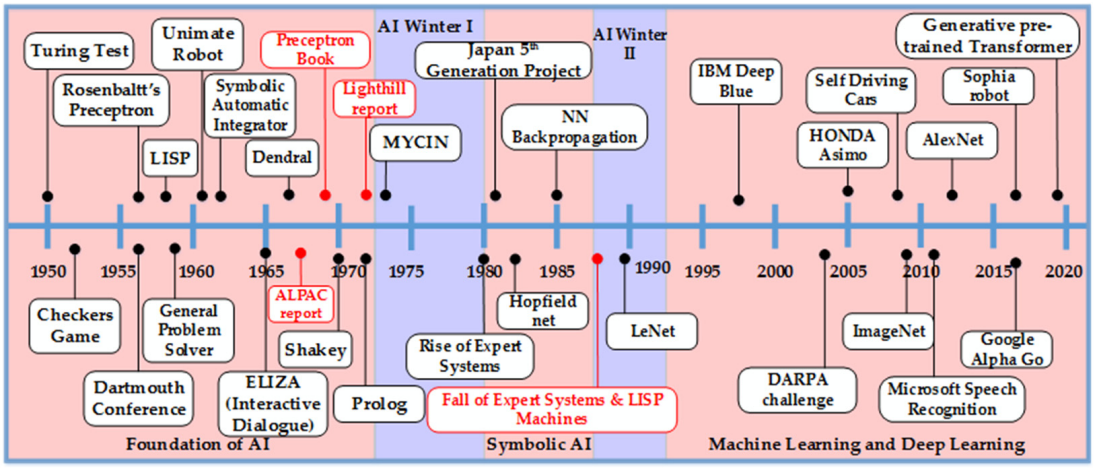

Figure 3. Artificial intelligence over the years.

3. Emergence of Microprocessors

The development of microprocessors is a direct consequence of the invention of semiconductor transistors by Bell Labs in 1947 and the creation of integrated circuit (IC) chips by Robert Noyce in 1961. In 1969, Busicom (Busicom, Osaka and Tokyo, Japan), a calculator company, contacted Intel to order 12 chips for their calculators. In return, Intel came up with four designs, and one of them could be programmed in different ways to fulfill the customers' requirements. Intel named it Intel 4004 and launched it along with its chipset in 1971. Intel 4004 [29] is known as the first microprocessor, though it was not very powerful, as it could only perform simple 4-bit arithmetic operations. Meanwhile, Texas Instrument (TI) also filed a patent on microprocessors, which was issued in 1973 [69]. TI's first microprocessor was introduced in 1974, with the name TMS1000, which was also a 4-bit microprocessor with 32 byte RAM and 1 KB ROM. It initiated an avalanche of research

and development in this field. Continued interest in improvement in IC technology led to a diverse range of 8- and 16-bit microprocessors within the next few years.

Intel 4004 was followed by the Intel 8008 and 8080 microprocessors. Intel 8080 was an 8-bit microprocessor and the first one to be a part of home-based personal computers. It was later updated to Intel 8085 by adding more instructions, interrupt, and serial input/output (I/O) pins. Motorola also developed its 8-bit microprocessor, the 6800 family, at about the same time. Motorola's 6800 (Motorola, Chicago, IL, United States) did not play a significant role in minicomputers, but made a great impact on the automotive market [70]. In this way a huge market for embedded processors in the automotive industry was established. The success of Intel 8080 and Motorola's 6800 led to the development of microcomputers like Atari 2600 (Atari Inc., Sunnyvale, CA, USA), Nintendo Entertainment System (Nintendo, Kyoto, Japan); Commodore 64 (Commodore International, West Chester, PA, USA), Zilog Z80 (Zilog, Milpitas, CA, USA) also came with a DRAM refresh signal and on-chip clock signals to provide better interfacing capability [44]. The increase in acceptance of metal oxide semiconductor (MOS) fabrication technology mainly drove the computer revolution in the 1980s.

The 8-bit architecture of Intel's 8085 and 8088 was improved to 16 bits in the form of the Intel 8086 microprocessor. The 8086 consisted of bus interface units (BIUs) and execution units (EUs), which were structured to carry out simple pipelined operations wherein the BIU fetched the instructions and EU processed them. The 8086 was also accompanied by a matching floating-point math co-processor chip, 8087, which was based on the implementation of IEEE floating point standard IEEE-754 [71]. Around the same time, the 16-/32-bit Motorola 68000 further advanced microprocessor architecture to fetch the instructions of one or more 16-bit words, which eventually paved the way for a full 32-bit architecture. Another Motorola processor, the 68010, introduced the concept of virtual memory. In 1984, Motorola's 68020 turned out to be first 32-bit microprocessor with an actual pipeline and an on-chip 256-byte instruction cache.

Thanks to these advancements in processor architecture, computers became accessible to the common person, albeit for use in business and entertainment. Although embedded applications also featured microprocessors, the major requirements of embedded devices in the 1980s and 1990s (calculators, watches, PID controllers, industry displays, etc.) were low cost and convenience of use. Their manufacturers could choose any low-cost microprocessor that gave the necessary performance for that specific application. On the other hand, PC users demanded more variety of applications on the same desktop and got accustomed to utilizing different software libraries. In this scenario, the performance of general-purpose microprocessors became one of the main concerns for customers. The release of continuous updated versions of operating system software and high-level languages, e.g., Windows, C++, Java, etc., also increased the emphasis on the technology of PCs. The Motorola 68000 offered an advanced architecture, but IBM adopted an 8-bit microprocessor, Intel 8088, to power its initial IBM PC in 1981, as it provided a better software interface and utilization with 8-bit peripherals [72]. Not only did IBM establish itself as a leader in the PC market, but also set benchmarks and standards for others.

The development of PCs laid the foundation of the new era for microprocessors. In addition, the evolving complementary metal oxide semiconductor (CMOS) technologies corroborated Moore's Law and played a vital role in giving shape to new architecture of microprocessors. Mead and Conway's work [73] on very large-scale integration (VLSI) introduced new design methodologies for academia and industry. Many computer-aided design (CAD) and simulation tools were created, e.g., those from Cadence [74], Synopsys [75], Mentor Graphics [76], etc., to draw schematics and analyze VLSI circuits at various performance levels [73]. It enabled the designers to test new design methodologies and evaluate their performance in the design stage to bring innovations to the architecture of microprocessors. The ever-shrinking geometry of CMOS also helped catalyze the evolution of microprocessors. The transistor channel length deceased to below 1.25 microns in 1985 [77], which resulted in Intel's 386DX microprocessor, with a gate length equal to 1 micron. It

became possible to integrate the entire CPU (excluding memory and floating-point units) on a single chip by the end of the 1980s. The previously used NMOS technology was overruled by CMOS due to its low power dissipation. At that point, the semiconductor vendors were the dominant microprocessor suppliers, who had pretty good knowledge of fabrication and chip making, but lacked intricate details about the internal processor architecture. The future complexities of architecture required specialized knowledge to shift the market to 16-bit architecture as well as embedded microprocessors.

After the mid-1980s, the emphasis shifted to the development of efficient and powerful 16-bit and 32-bit microprocessors as well as corresponding technologies for memory storage and interconnections. Previously, VAX 11/780 by DEC had been a prominent 32-bit mini-computer, but its architecture was based on smaller multi-chip processors. On the other hand, Patterson and Ditzel's work [31] introduced a new paradigm, called reduced instruction set computer (RISC) architecture, that provided a possible solution to optimize the hardware within given resources as compared to the multi-chip idea of VAX 11/780. The project at University of California, Berkeley, also supported the idea of RISC by designing the Barkley RISC I/II processors [78]. The RISC setup added proper pipelined architecture to microprocessors. For example, RISC I and II used two-stage and three-stage pipelining, respectively. John Cocke's IBM 801 minicomputer also reinforced RISC concepts on other types of computer organizations [79]. Later, Stanford University took architecture optimization to a new level by introducing a microprocessor without interlocked pipeline stages. They called this millions of instructions per second (MIPS) architecture [80].

Contrary to the previously used complex instruction set computer (CISC) architecture such as Intel 80386, RISC architecture supported 32-bit fixed-length instructions, larger general-purpose registers, and pipelined stages, and avoided memory-to-memory instructions. The simple and regular instruction set for RISC obviated micro-coded ROM, which created more space for full 32-bit instructions. Initially, only smaller OEM accepted the argument about RISC; Acorn Computer in the UK was one of them. They started by amending previously emerging 16-bit microprocessors but were hindered by two major problems: (i) real-time performance in I/O handling and (ii) memory bandwidth utilization. They built their own 32-bit design popularly known as the Acorn RISC Machine (ARM) to overcome these drawbacks [45].

Driven by promising claims of RISC architecture, the first commercial RISC CPU, MIPS R2000, entered the market by the mid-1980s [44]. Intel and Motorola also started the development of their own RISC-based microprocessors. A year after the introduction of Motorola 68020, Intel also built its first 32-bit microprocessor, the 80386DX. Motorola 68020 and Intel 80360DX constituted a limited number of pipelined stages. Intel modified its first-generation microprocessors (8086, 80286) by adding new modes of memory addressing, more instructions, and an on-chip memory management unit (MMU). Three years later, Motorola developed 68030, with an integrated MMU and dynamic bus size selection [81]. The increasing number of transistors in microprocessors deepened the pipeline to five stages, as shown in Figure 4, and also incorporated caches and their control functions, the MMU and floating-point unit (FPU), on a single chip. In 1989, Intel used 1.2 M transistors in 80486DX (as compared to 275K in 803868DX) with on-chip FPU [82].

The RISC philosophy and increasing transistor density elucidated the architecture of microprocessors by adding deeper pipelined stages, more on-chip functional units and multilevel caches, higher issue rates, and wider bandwidths [83]. The design of microprocessors was further driven by the race of higher clock rates. It was believed that increasing the clock frequency would directly increase the performance, in line with Moore's Law. Alpha 21064 was among the first microprocessors to attain a frequency of 150 MHz, and it was followed by Alpha 21164 with 500 MHz [84]. In competition, Motorola, IBM, and Apple collectively designed PowerPC, with a clock speed of 233 MHz. PowerPC adopted a more balanced design scheme as compared to the Alpha series, which compromised the number of instructions and latency for higher clock rates [84]. The desktop market was overtaken by the 32-bit Superscalar Intel Pentium, which sped up

many PC-based applications, and its performance limits were further stretched by super-pipelined Pentium Pro. In later years, many versions of the Pentium series were introduced, leading to Pentium III, which pushed the operating frequency beyond 1 GHz [85]. Another company, AMD, giving competition to Intel, developed its first processor, working at 75 to 133 MHz, in 1996 [86]. AMD also jumped in the race of clock rates and released many successors with higher clock speeds. A major advancement was AMD's Athlon processors, which supported the first 64-bit data path in 2003, and with that the world of microprocessors had reached a new level of performance [86]. At the same time, the embedded market was also developing simultaneously. ARM was quick to cater to this emerging market and established itself as a leader in the mobile handset area through the use of a 16-bit thumb instruction set [87] and easy integration in a system on a chip (SoC) [88]. Table 1 describes some basic features of the popular superscalar architectures of the 1990s and early 2000s.

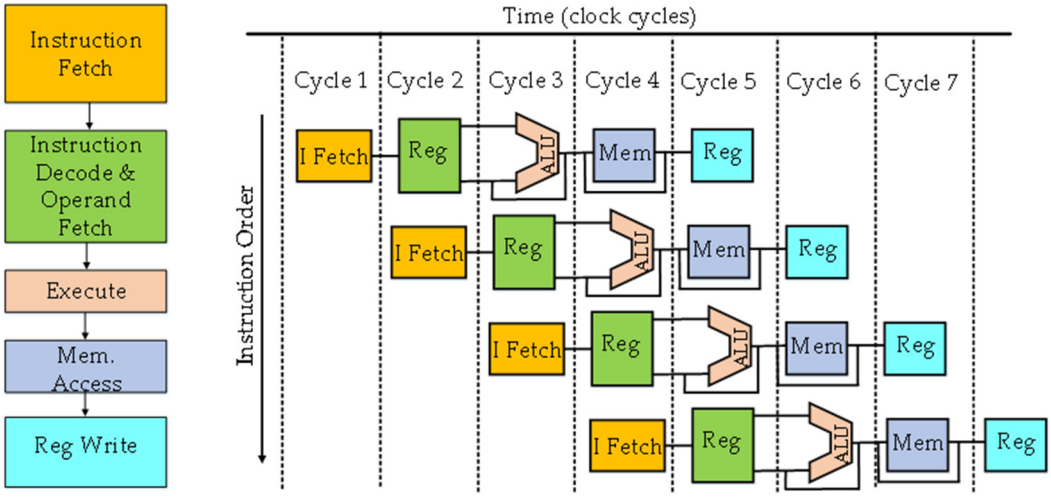

Figure 4. Five stages of pipelining in microprocessors.

Table 1. Basic features of popular superscalar microprocessors (1990s–2000s).

High-Performance (Superscalar) Microprocessors					
Microprocessor	**Year**	**Clock Speed (MHz)**	**Transistor Size (microns)**	**Cache Size (KB)**	**Pipe Stages**
Intel 486 (Intel, Santa Clara, CA, USA)	1989	25 to 50	0.8–1	8	5
Intel Pentium Pro (Intel, Santa Clara, CA, USA)	1995	200	0.35–0.6	8/8	12–14
DEC Alpha 21164 (DEC, Maynard, MA, USA)	1996	500	0.5	8/8/96	7
Power PC 604e	1997	233	0.25	32/32	6
AMD K5 (AMD, Santa Clara, CA, USA)	1996	75–133	0.35–0.5	8/16	5
MIPS R10000 (MIPS Technologies, Sunnyvale, CA, USA)	1996	200	0.35	32/32	5
Intel Pentium IV (Intel, Santa Clara, CA, USA)	2000	1400–2000	0.18	256	20

In the mid-2000s, the interest in the continuous increase in clock frequency was diminished because of power dissipation barriers. Both Intel and AMD strived for smaller feature sizes to achieve high operating frequency, but the chips became too hot and demanded impractical cooling systems for such high transistor densities. Figure 5 represents the predicted trend in heat dissipation levels due to the increase in the power density of Intel

chips. Surprisingly, if the chips continued to be manufactured with the same increasing densities, then they could have reached the power dissipation level of a rocket nozzle [89].

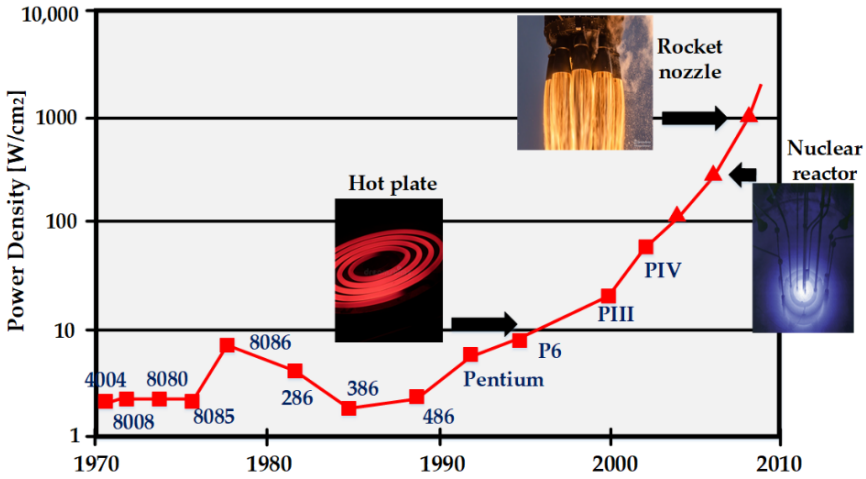

Figure 5. Trend of heat dissipation with the increase in power density of Intel chips.

Later, the performance bottleneck in microprocessors was overcome by adopting parallel computing techniques using closely coupled multiple processor cores on a single chip [90]. In 2005, AMD released its first dual core processor, Athlon 64 X2 [86]. Intel also rushed into multi-core design and came up with the Core-2 Duo processor for desktop and laptop computers. Some processors, like the Intel Core-2 Duo, were designed with a shared level-2 (L2) cache, whereas Intel Pentium-D and AMD Opteron came with a private L2 cache for each core or processor [91]. Multi-processor chip (MPC) architecture further improved to quad and more cores and initiated a new era of performance gains that yielded profitable results in terms of cost, power dissipation, and a corresponding range of new applications [92].

The Intel Core 2 Quad series enhanced the speed technology of desktop and mobile processors by incorporating four cores with greater sizes of cache memory on a single chip. The family of Core 2 Quad was followed by generations of high-end performance processors, Intel Core i-7. The grouping of Core i7 was usually based on its microarchitecture, with the first generation designed on 45 nm technology, and the recent 10th generation is based on 14 nm processing for up to 8-core chips. AMD also launched a series of processors under the name of "Ryzen" in 2017, which mainly featured the 8-core and 16-thread designs. In the following years, new generations of AMD Ryzen, built on advanced semiconductor technology, came to improve the performance, but significant improvement was witnessed in the third generation of the Ryzen brand [93], which was designed using 7 nm technology. In 2020, the Ryzen 5000 series crossed the Ryzen III gen's performance by providing a 19% increase in instructions per cycle (IPC) on the same technology. Intel marketed the Xeon family of processors, which mainly targets the workstation, servers, and embedded devices. The AMD Ryzen Threadripper beat the Intel in core count by incorporating a maximum of 64 cores on 7 nm technology. The Intel Xeon processors are designed using same architecture as desktop processors, but support higher number of cores, larger RAM size, advanced cache memory, and many other useful features. Table 2 summarizes the basic features of some of the most prominent multicore processors.

In tandem, the developments in cache memories, storage technologies, and faster interconnects have provided a support structure that enables microprocessors to provide maximum utilization of advances in their internal architecture to end-user applications [94]. Figure 6 shows the trends in the main characteristics of popular general-purpose CPUs over time. As this paper is aimed at discussing processor architectures, we will not digress, but it is important to mention the significant improvements in associated technologies that allow full performance gains to be obtained from microprocessors.

Table 2. Basic features of popular multicore microprocessors (2005 onwards).

Multicore Microprocessors					
Microprocessor	**Year**	**Clock Speed (GHz)**	**Transistor Size (nm)**	**Caches (MB)**	**Cores**
AMD Athlon 64 X2	2005	2	90–65	0.5	2
Intel Core 2 Duo	2006	2.66	65	4	2
Intel Core 2 Quad Q6600	2007	2.4	65	8	4
Intel Core i7-3770	2012	3.4	22	8	4
AMD Ryzen 7 1700x	2017	3-3.6	14	4/16	8
Intel Core i9 10900	2020	5.20	14	20	10
Intel Xeon Platinum 9282	2019	3.8	14	77	56
AMD Ryzen Threadripper 3990X (AMD, Santa Clara, CA, USA)	2020	4.3	7	32/256	64

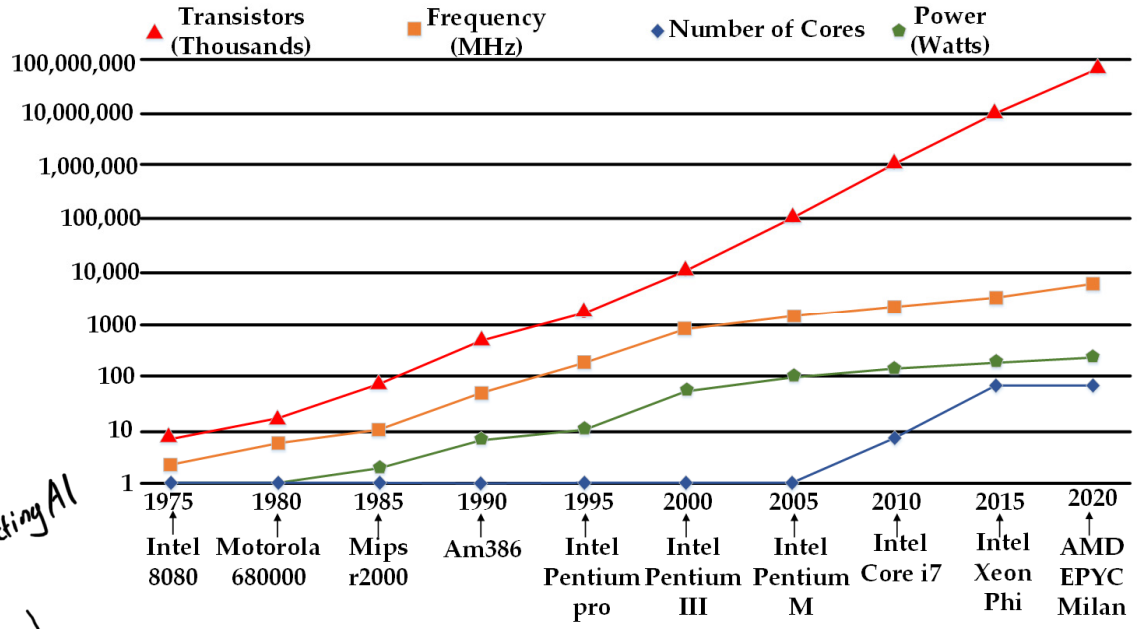

Figure 6. Performance evolution of general-purpose microprocessors.

How is this affecting AI

Hardware that were used to shallow neural network

Large Information helped provide data for training AI ?

4. Confluence between Artificial Intelligence and Microprocessors

As microprocessors evolved to cater to more complex processing and application domains, the field of AI was also able to overcome many constraints and provide implementable solutions for different challenging tasks. The first practical AI implementation was a handwritten digit recognizer, LeNet [95]. It led to the development of hardware for shallow neural networks, for example, Intel ETANN [96,97], SYNAPSE-1 [98], and ANNA [99]. The AI field flourished under the banner of machine learning (ML), and many self -learning algorithms were proposed to classify images, categorize text, and recognize hand-written characters [100–102]. Within the ML domain, neural networks (NNs) became the most popular choice because of their accuracy and straightforward implementation. Deep learning (DL) enhanced the multi-layered architecture of NNs by extracting features at new levels of abstractions [13]. A major breakthrough for DL was seen in the early 2010s, when a large amount of information was incorporated for the training of AI models such as MNIST [103] and CIFAR [104]. The ImageNet dataset was launched about the same time, with more than 3 million labeled images of different categories [105]. Microsoft presented its speech-recognition system based on deep neural networks in 2011 [106]. Further exemplary performances of DL were illustrated by the ImageNet annual challenge [107].

In 2012, the error rate of ImageNet classification dropped to approximately 15% from 25% by utilizing a convolution neural network (CNN)-based model, AlexNet [108]. The classification error rate was further decreased to only a few percent in later years. This success for CNNs in AI implementation shifted the research direction to DL and inspired the development of different hardware platforms for its implementation.

Even though the speed of sequential microprocessors increased drastically over the years, it could only improve performance to a certain extent. The massive computational demands of early AI networks could not be fulfilled by available microprocessors, as some of the AI models needed more than 1 giga floating point operations per second (GFLOPS) to be able to execute in real time [109]. The advancement in the processing throughput of microprocessors came about mainly due to the increase in frequency and most importantly due to the enhanced increase of parallelism. The parallel computation was exploited through multiple instructions issue through pipelining as well as superscalar architectures [90]. Thread level parallelism (TLP) was introduced using multiple processors on a single chip [90]. The boost in performance obtained by multicore processors took the performance gains to the next level, which paved the way for real-time processing of AI applications. Figure 7 shows the comparison between CPU operations per second for single-core Intel Pentium IV 2.4 GHz and Intel Pentium IV 2.8 GHz with dual-core Intel Pentium G640T 2.4 GHz [110]. Intel Core 2 Extreme quad-core provided impressive performance for AI-based applications, especially for gaming on desktops [92]. The multilevel cache memories also contributed to the deployment of AI by providing a faster way to access the memory for huge AI models and large datasets.

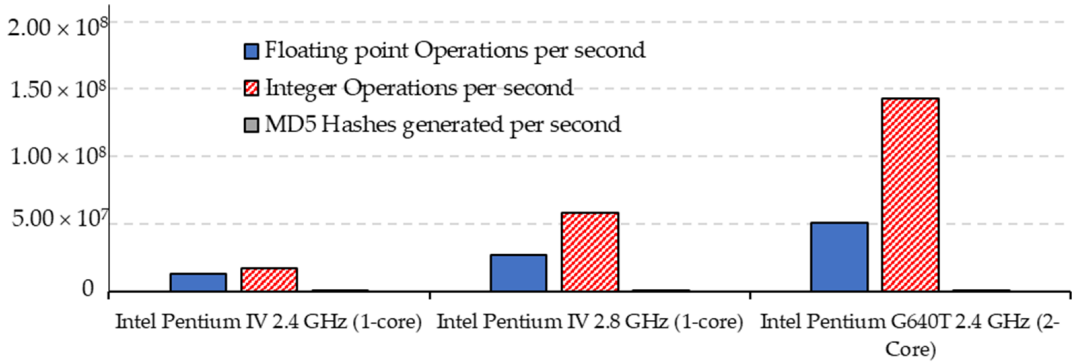

Figure 7. Comparison between CPU operations per second for single core and dual core.

The most demanding calculation in CNN computation is the multiply and accumulate (MAC) operation between input and trained weights, which can be performed on multiple data elements simultaneously to achieve faster execution. To facilitate the NN processing, many modern processors feature a special vector instruction set to employ single instruction, multiple data (SIMD). Intel included advanced vector extensions, a 256-bit vector (Intel AVX-256), for each core, which could support the processing of eight single-precision (32-bit) floating point (FP) operations or four double-precision FP operations in a single instruction. Later, Intel AVX-512 increased the size of the vector unit to 512 bits, which doubled the FP operations per instruction [111]. Many updated Intel Xeon processors, like the Xeon Phi series, supported Intel AVX-512 architecture, where they had a designated vector unit per core, as shown in Figure 8 [112]. ARM also launched the ARM Scalable Vector Extension (ASVE) in their modern processors like the ARM Neon [113]. The parallelization technique is further enhanced by mapping fully connected (FC) and convolution (Conv) layers of NNs into matrix multiplications [114,115]. Figure 9 represents the mapping of FC layers as a matrix-vector multiplication with one input feature map with C × H × W = Input_Channels × Height × Width and matrix–matrix multiplication for N number of input feature maps. Figure 10 shows the mapping and arrangement of a Conv layer onto matrix–matrix multiplications. Several compatible optimized software libraries were also

launched to support matrix multiplication for DNNs running on these high-performance architectures. Open BLAS (basic linear algebra subroutines) is for multiple microprocessors like ARM, Intel, and MIPS. Intel also developed the Math Kernel Library (MKL) for its processors [116]. There is another popular deep learning framework, called Caffe, for Intel processors [117]. Many tech companies like Facebook and Google have contributed to optimizing other software technologies for Intel architecture for the flexible implementation of modern DNNs.

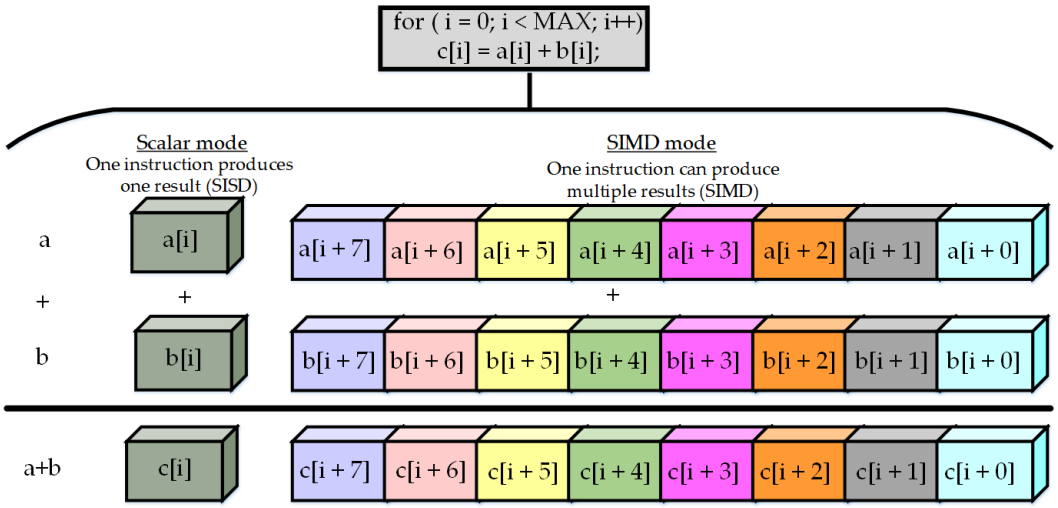

Figure 8. Single-instruction multiple-data (SIMD) operation.

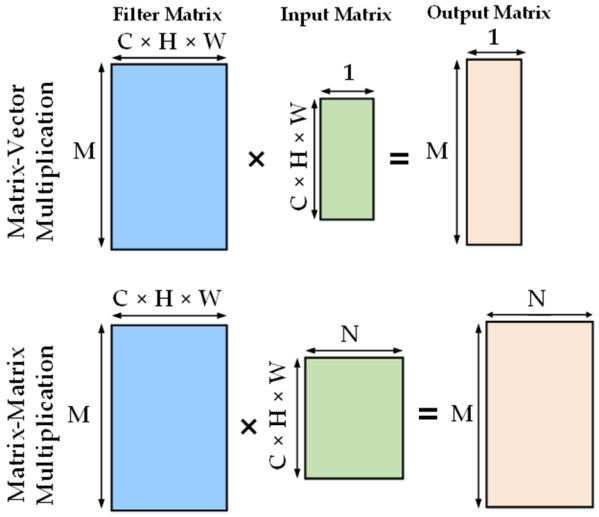

Figure 9. Mapping of fully connected (FC) layers onto matrix multiplication.

Intel built Xeon Scalable processors specialized in running complex AI architectures efficiently along with other workloads using the Intel Deep Learning Boost (DL Boost) library. It extended the Intel AVX-512 with Intel inference boost instructions to optimize the vector neural network instructions (VNNI) for deep learning inference. This brought significant performance improvement to image classification, language translation, object detection, speech recognition, and other AI applications in mobile phones, desktops, and servers [118]. Intel DL Boost sped up AI processing by a factor of two, as it used a single int-8 instruction to handle DL convolution, which was previously using three AVX-512 instructions [118]. Previously used 32-bit single-precision FP instructions were converted to 16-bit integer instructions for training and INT8-bit for the inference of DL models with negligible loss in accuracy. Memory access is a bottleneck in the processing of DNNs and also requires a higher order of energy compared to the computational workload [119]. The

lower numerical precision not only led to the reduction of memory bandwidth but also helped in efficient utilization of cache memories. Furthermore, it also increased the overall computational throughput [120]. The Intel DL Boost incorporated the brain floating-point format (bfloat16), which enabled a dynamic range of numerical values using a floating radix point [121]. The bfloat16 data type is also included in ARM [122] and AMD [123] microprocessors.

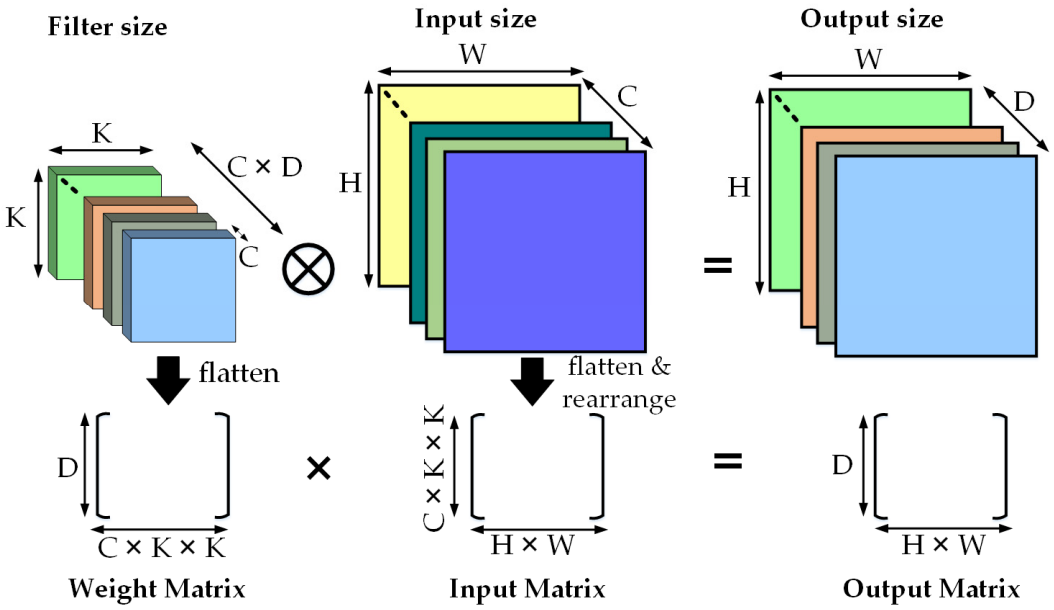

Figure 10. Mapping of convolution (Conv) layers onto matrix multiplication.

As mentioned above, NN computation is highly parallel in nature, whereas the general-purpose CPUs deal with a wide variety of sequential applications, e.g., binary search trees, data retrieval, string matching, etc. Consequently, graphic processing units (GPUs), which are inherently parallel, became the warehouse for AI processing. Initially, GPUs were used for computer graphics, which is a highly parallel application. The revolution in the architecture of GPUs enabled its use in general-purpose applications. GPUs work efficiently for massively parallel algorithms like AI because of the integration of hundreds (recently, thousands) of cores into one chip [38]. Nvidia realized this potential of GPUs and developed a software library called Compute Unified Device Architecture (CUDA) along with compatible hardware architectures [38]. The Nvidia V100 architecture has 5120 × 32-bit floating-point cores and 2560 × 64-bit floating-point cores, whereas the Intel Xeon Phi family includes 64 to 72 general-purpose cores [124], as mentioned in Table 3 [125]. Due to high computing requirements, AlexNet [108] was implemented with GPUs for the processing of 61 million weights and 724 million MAC operations. It led to the evolution of deeper architecture for convolution networks. A popular DNN called Overfeat adopted the architecture of AlexNet but with a greater number of arithmetic operations (2.8 giga MACs per image) [126]. Another DNN, VGG-16, saw a further increase in the number of weights and MACs of up to 138 million and 15.5 giga operations, respectively [127]. The GoogLeNet comes with a 22-layered architecture with an inception module [128] and it is designed to store all the trained weights in a GPU memory. The multidimensional filters are key to extracting the useful pattern or features of the input data in CNNs, and GoogLeNet uses the filter size of 1 × 1 to reduce the number of weights [129]. There are many updated versions of GoogleLeNet with increased accuracy and corresponding computing cost [130,131].

Table 3. Comparison of the number of CPU and GPU cores.

CPU			GPU		
Processors	**Minimum Cores**	**Maximum Cores**	**Processors**	**Tensor Cores**	**CUDA Cores**
Intel Core i7, 10th Gen	4	8	**Nvidia RTX 2080**	-	4352
AMD Ryzen	4	16			
Intel Core i9, 10th Gen	8	28	**Nvidia V100**	640	5120
Intel Xeon Plat. I Gen	4	28			
Intel Xeon Plat. II Gen	4	56	**Nvidia A100**	432	6912
AMD Ryzen Threadripper	24	64			

ResNet went even deeper, with 34 layers [132], and it was the first one to achieve an error rate of less than 5% in the ImageNet challenge. Nvidia's GPU allowed the implementation of such complex NNs using popular DL frameworks like PyTorch [133], Caffe [117], and Tensorflow [134] through the use of the CuDNN [135] library. The CuDNN library belongs to CUDA-X AI [136], which is a collection of libraries for Nvidia GPUs to accelerate DL and ML. Nvidia's latest two GPUs (V100 and A100) [137,138] were built with a combination of traditional CUDA and tensor cores. The tensor cores specialize in accelerating the large mixed-precision Matrix MAC operations in a single instruction. The pairing of CUDA and tensor cores enables the Tesla V100 architecture to deliver 120 TFLOPs for DL [137]. The Nvidia A100 GPU enhanced the performance by increasing the number of cores (see Table 3) and supporting numerical formats like INT4, TF32, and others. Tensor format TF32 is a new format to accelerate 32-bit floating-point instructions up to 10 times faster than the V100 32-bit floating-point instruction in DL frameworks. This was further improved 2× by adding a new feature of sparsity in tensor cores [138]. The A100 sparsity pruned the trained weights with the supported sparse pattern and by making an efficient hardware architecture to process the trained weights [138].

Recently, technology has been heading towards a dedicated hardware platform for application-specific AI processing. Many DNN accelerators have been proposed and implemented on FPGAs [139] and ASIC [40–42] and are usually based on a spatial architecture, as shown in Figure 11. The spatial architecture is designed using dataflow processing through a connected array of processing engines (PEs) (a combination of ALU and its local memory). This two-dimensional (2D) interconnection of PEs facilitates the reuse of the data to reduce the frequency of memory access and increase the parallel computation. The data from the memory can flow left to right and top to bottom through the array of PEs. The PEs are assigned to perform MAC operations on the coming data in a specific manner depending on the dataflow technique.

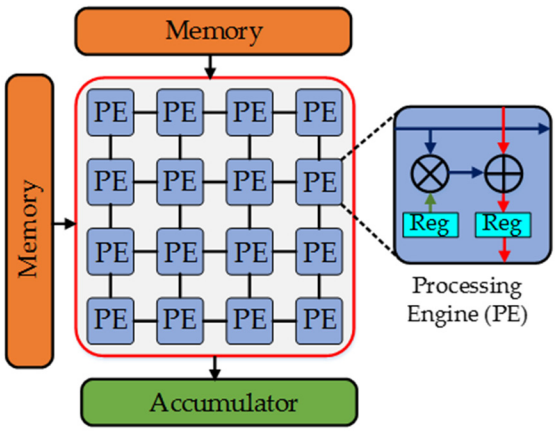

Figure 11. Hardware block diagram showing the generic structure of a spatial architecture.

The data-handling techniques in a spatial architecture can be classified as:

- Weight Stationary (WS): The weights are kept fixed in PEs while inputs flow through the array of PEs with the movement of partial sums. An example includes neuFlow [140] and others [43,141].
- Output Stationary (OS): The accumulation of partial sums is kept constant in PEs to minimize the energy consumption of reading and writing partial sums while broadcasting the inputs and weights to the array of PEs just like in ShiDianNao [142].
- No Local Reuse (NLR): Nothing stays stationary, as the local memory for PEs is eliminated to reduce the area requirement. For instance, DianNao [143] has an NLR dataflow.
- Row Stationary (RS): It aims to minimize the memory access cost by reusing all types of data (weights, inputs, and partial sums) by mapping the rows of convolution on PEs for each sliding window. Eyeriss [144] is one of the accelerators based on RS architecture.

The highlight among all these is an industrial platform called Tensor Processing Unit (TPU), developed by Google. The first TPU was deployed in the Google data center in 2015 [43]. It consisted of systolic arrays of PEs designed for WS dataflow that resembles 2D SIMD architecture. It was followed by another TPU that could process both the training and inference of DNNs in the data center [145]. Later, Google also launched its "edge TPU" for inference in Internet of Things (IoT) applications [146].

5. Future Roadmap and Challenges

In the past decade, AI has grown rapidly in its performance and range of applications. It has affected every industry and every human directly or indirectly. Many real-life applications have integrated AI into their functionalities to give exceptional benefits. Still, we are at the beginning of AI in many practical fields. As observed by the current trend in higher accuracy afforded by DNNs, AI algorithms will continue to go deeper into neural structure to attain the capabilities of the human brain to precisely handle critical tasks [147]. Many companies like Apple and Google have huge budgets dedicated to the progress of AI, and academic institutions are also recognizing AI as a distinct field of learning [148]. In this scenario, AI is expected to progress in new and more innovative directions. AI has already given a tremendous boost to several upcoming fields like IoT, big data, autonomous vehicles, and intelligent robotics, and it will continue to drive these technologies in the future. The ongoing revolution of AI will only see an improved uptake in the near future.

Computing capabilities have always been a challenge to the progress of AI. The main focus is on hardware platforms to manage sufficient resources that will be able to fulfill the growing demands of AI. Most of the progress in microprocessors to date has been due to a reduction in transistor size (courtesy of Moore's Law), but it has already deviated from the predicted performance path, as thermal issues become unavoidable after a certain clock limit. Still, microchips are designed on a very thin layer of silicon wafer, but there some ideas revolving around the three-dimensional structure of microchips to increase the efficiency of microprocessors [149]. Along with benefiting high performance, this concept raises many thermal and interconnectivity issues that researchers need to address before its successful adoption. In other words, microprocessors will prevail with the current trends in architecture for another decade. At the same time, we cannot deny the possibility of novel techniques like quantum computing and molecular computing to change the design of future microchips [150].

Microprocessors have also attained a new level of performance and efficiency due to various parallelization and vector processing techniques that proved to be a driving tool for AI. Considering the limitation of parallel hardware in microprocessors and the deviation from Moore's Law, it is predicted that the next generation computers will be based on microprocessors working along with highly specialized accelerators dedicated to processing power-hungry AI algorithms. The current research field of AI accelerators will continue to grow and evolve with more advanced trends to make finely tuned accelerators

for AI [151]. TPUs are a successful example of application-specific hardware, but it will take some time to adapt the application-specific approach to smart computers, mobile phones, and other embedded applications.

There were huge expectations from AI to change the shape of daily lives, but privacy and security issues are now becoming a major concern in the contemporary era of IoT and big data. There are already hackers out there attacking sensitive industries and data [152,153]. These attacks are expected to increase with the deeper penetration of AI in society through the adoption of smart cities, autonomous vehicles, and intelligent industrial machines. To increase trust in machine intelligence, there is a need for high-performance hardware-enhanced secure technologies. Overall, looking at the evolution of AI since the 1950s, it can be foreseen that the AI field will surmount all these challenges and evolve further in the future with the help from the increasing capabilities of microprocessors.

6. Conclusions

The idea of artificial intelligence was initiated way before the birth of microprocessors. For the first 30 years, most AI work was at the algorithmic level. The advent of microprocessors created the need for AI machines, but AI was simultaneously evolving in different directions, and this sluggish progress in the initial years made scientists lose interest in its widespread acceptance. Thus, AI has seen many downfalls during its evolution, but every time it has risen again with new hope and promise. In parallel, the microprocessor was progressing with its dynamics governed by Moore's Law and MOS technology. The advancements in microprocessor architecture significantly improved its performance and efficiency. The introduction of RISC architecture, superscalar, deep pipelining, and multicore designs gave an exponential boost to the computing power of microprocessors. AI developers soon realized the available computing capability and started to make inroads with machine learning, deep learning and associated datasets. After witnessing the promising results of AI in various applications, microprocessors started supporting AI by amending their architecture in such a manner that they could execute the complex algorithms of AI efficiently for different tasks in the form of SIMD, GPUs, and TPUs. Since then, the two fields, computer architecture and AI, have complimented each other, as microprocessors have adopted different techniques to fuel the growing demands of complex AI models. This confluence between both fields brings AI to every home and industry through PCs, smart gadgets, and other embedded platforms. It can be safely predicted that AI and microprocessor architecture will continue to evolve together in the future with new topologies and dedicated accelerators to deal with challenges like data security and information complexity of data-intensive applications like big data.

Author Contributions: Conceptualization, M.A.P. and S.M.; methodology, F.H.K., M.A.P. and S.M.; writing—original draft preparation, F.H.K., M.A.P. and S.M.; writing—review and editing, F.H.K., M.A.P. and S.M.; supervision, M.A.P. and S.M.; All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Turing, A.M. Computing Machinery and Intelligence. *Mind* **1950**, *59*, 433–460. [CrossRef]
2. Newell, A.; Shaw, J.C.; Simon, H.A. Report on General Problem-Solving Program. In Proceedings of the International Conferenceon Information Processing, Paris, France, 30 December 1958.
3. Gelernter, H.L.; Rochester, N. Intelligent Behaviour in Problem- Solving Machines. *IBM J. Res. Dev.* **1958**, *2*, 336. [CrossRef]
4. Weizenbaum, J. ELIZA—A Computer Program for the Study of Natural Language Communication between Man and Machine. *Commun. Assoc. Comput. Mach. (ACM)* **1966**, *9*, 36–45. [CrossRef]

5. Rosenblatt, F. The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain. *Psychol. Rev.* **1958**, *65*, 386–408. [CrossRef] [PubMed]
6. Delipetrev, B.; Tsinaraki, C.; Kostic, U. *Historical Evolution of Artificial Intelligence*; EUR 30221 EN; Publications Office of the European Union: Luxembourg, 2020.
7. Nikolopoulos, C. *Expert Systems: Introduction to First and Second Generation and Hybrid Knowledge Based Systems*; Marcel Dekker, Inc.: New York, NY, USA, 1997.
8. Shortliffe, E.H.; Davis, R.; Axline, S.G.; Buchanan, B.G.; Green, C.C.; Cohen, S.N. Computer-based Consultations in Clinical Therapeutics: Explanation and Rule Acquisition Capabilities of the MYCIN System. *Comput. Biomed. Res.* **1975**, *8*, 303–320. [CrossRef]
9. Shafique, M.; Theocharides, T.; Bouganis, C.S.; Hanif, M.A.; Khalid, F.; Hafız, R.; Rehman, S. An overview of next-generation architectures for machine learning: Roadmap, opportunities and challenges in the IoT era. In Proceedings of the 2018 Design, Automation & Test in Europe Conference & Exhibition (DATE), Dresden, Germany, 19–23 March 2018.
10. Khan, F.H.; Ashraf, U.; Altaf, M.A.B.; Saadeh, W. A Patient-Specific Machine Learning based EEG Processor for Accurate Estimation of Depth of Anesthesia. In Proceedings of the 2018 IEEE Biomedical Circuits and Systems Conference (BioCAS), Cleveland, OH, USA, 17–19 October 2018.
11. Cao, B.; Zhan, D.; Wu, X. Application of SVM in Financial Research. In Proceedings of the 2009 International Joint Conference on Computational Sciences and Optimization, Sanya, China, 24–26 April 2009.
12. He, K.; Zhang, X.; Ren, S.; Sun, J. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015.
13. LeCun, Y.; Bengio, Y.; Hinton, G. Deep Learning. *Nature* **2015**, *521*, 436–444. [CrossRef] [PubMed]
14. Wiafe, I.; Koranteng, F.N.; Obeng, E.N.; Assyne, N.; Wiafe, A.; Gulliver, A.S.R. Artificial Intelligence for Cybersecurity: A Systematic Mapping of Literature. *IEEE Access* **2020**, *8*, 146598–146612. [CrossRef]
15. Vishnukumar, H.J.; Butting, B.; Müller, C.; Sax, E. Machine learning and Deep Neural Network—Artificial Intelligence Core for Lab and Real-world Test and Validation for ADAS and Autonomous Vehicles: AI for Efficient and Quality Test and Validation. In Proceedings of the Intelligent Systems Conference (IntelliSys), London, UK, 7–8 September 2017.
16. Hashimoto, Y.; Murase, H.; Morimoto, T.; Torii, T. Intelligent Systems for Agriculture in Japan. *IEEE Control Syst. Mag.* **2001**, *21*, 71–85.
17. Khan, F.H.; Saadeh, W. An EEG-Based Hypnotic State Monitor for Patients During General Anesthesia. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.* **2021**, *29*, 950–961. [CrossRef]
18. Mohsen, H.; El-Dahshan, E.-S.A.; El-Horbaty, E.-S.M.; Salem, A.-B.M. Classification using Deep Learning Neural Networks for Brain Tumors. *Future Comput. Inform. J.* **2018**, *3*, 68–71. [CrossRef]
19. Ying, J.J.; Huan, P.; Chang, C.; Yang, D. A preliminary study on deep learning for predicting social insurance payment behavior. In Proceedings of the IEEE International Conference on Big Data (Big Data), Boton, MA, USA, 11–14 December 2017.
20. Zanc, R.; Cioara, T.; Anghel, I. Forecasting Financial Markets using Deep Learning. In Proceedings of the IEEE 15th International Conference on Intelligent Computer Communication and Processing (ICCP), Cluj-Napoca, Romania, 5–7 September 2019.
21. Grigorescu, S.; Trasnea, B.; Cocias, T.; Macesanu, G. A survey of Deep Learning Techniques for Autonomous Driving. *J. Field Robot.* **2020**, *37*, 362–386. [CrossRef]
22. Huang, P.-S.; He, X.; Gao, J.; Deng, A.A.L.; Heck, L. Learning Deep Structured Semantic Models for Web Wearch using Clickthrough Data. In Proceedings of the 22nd ACM international conference on Information & Knowledge Management (CIKM '13), New York, NY, USA, 27 October–1 November 2013.
23. Dahl, G.E.; Sainath, T.N.; Hinton, G.E. Improving Deep Neural Networks for LVCSR using Rectified Linear Units and Dropout. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, BC, Canada, 26–31 May 2013.
24. Zhang, D.; Liu, S.E. Top-Down Saliency Object Localization Based on Deep-Learned Features. In Proceedings of the 11th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI), Beijing, China, 13–15 October 2018.
25. Quillian, M.R. The Teachable Language Comprehender: A Simulation Program and Theory of Language. *Assoc. Comput. Mach.* **1969**, *12*, 459–476. [CrossRef]
26. Details for Component Intel Core i7-8086K. SiSoftware Official Live Ranker. Available online: https://ranker.sisoftware.co.uk/show_device.php?q=c9a598d1bfcbaec2e2a1cebcd9f990a78ab282ba8cc7e186bb96a781f3ceffd9b08dbd9bf3cefed8a09dad8bee8bb686a0d3eed6&l=en (accessed on 6 April 2021).
27. Cray-1 Computer System Hardware Reference Manual 2240004. Cray Research, Inc., 4 November 1977. Available online: http://bitsavers.trailing-edge.com/pdf/cray/CRAY-1/2240004C_CRAY-1_Hardware_Reference_Nov77.pdf (accessed on 6 April 2021).
28. Xiao, L.; Bahri, Y.; Sohl-Dickstein, J.; Schoenholz, S.S.; Pennington, J. Dynamical Isometry and a Mean Field Theory of CNNs: How to Train 10000-layer Vanilla Convolutional Neural Networks. 2018. Available online: http://arxiv.org/abs/1806.05393 (accessed on 6 April 2021).
29. Intel's Museum Archive, i4004datasheet. Available online: http://www.intel.com/Assets/PDF/DataSheet/4004_datasheet.pdf (accessed on 10 April 2021).

30. iAPX 86, 88 USER'S MANUAL. August 1981. Available online: http://www.bitsavers.org/components/intel/_dataBooks/1981_iAPX_86_88_Users_Manual.pdf (accessed on 4 June 2021).
31. Patterson, D.A.; Ditzel, D.R. The Case for the Reduced Instruction Set Computer. SIGARCH Comput. *Archit. News* **1980**, *8*, 25–33. [CrossRef]
32. History of the Development of the Arm Chip at Acorn. Available online: https://www.cs.umd.edu/~{}meesh/cmsc411/website/proj01/arm/history.html (accessed on 6 April 2021).
33. Moore, G. Cramming more Components onto Integrated Circuits. *Electronics* **1965**, 114–117. [CrossRef]
34. Hennessy, J.L.; Patterson, D.A. *Computer Architecture: A Quantitative Approach*; Morgan Kaufman Publishers, Inc.: San Mateo, CA, USA, 2012.
35. Seto, K.; Nejatollah, H.; Kang, J.A.S.; Dutt, N. Small Memory Footprint Neural Network Accelerators. In Proceedings of the 20th International Symposium on Quality Electronic Design (ISQED), Santa Clara, CA, USA, 6–7 March 2019.
36. Guo, X.; Ipek, E.; Soyata, T. Resistive Computation: Avoiding the Power Wall with Low-Leakage, STT-MRAM based Computing. *SIGARCH Comput. Archit.* **2010**, *38*, 371–382. [CrossRef]
37. Wulf, W.A.; McKee, S.A. Hitting the memory wall: Implications of the obvious. *SIGARCH Comput. Archit.* **1995**, *23*, 20–24. [CrossRef]
38. Baji, T. Evolution of the GPU Device widely used in AI and Massive Parallel Processing. In Proceedings of the 2018 IEEE 2nd Electron Devices Technology and Manufacturing Conference (EDTM), Kobe, Japan, 13–16 March 2018.
39. Deng, L.; Li, G.; Han, S.; Shi, L.; Xie, Y. Model Compression and Hardware Acceleration for Neural Networks: A Comprehensive Survey. *Proc. IEEE* **2020**, *108*, 485–532. [CrossRef]
40. Chen, Y.; Luo, T.; Liu, S.; Zhang, S.; He, L.; Wang, J.; Li, L.; Chen, T.; Xu, Z.; Sun, N.; et al. DaDianNao: A Machine-Learning Supercomputer. In Proceedings of the 47th Annual IEEE/ACM International Symposium on Microarchitecture, Cambridge, UK, 13–17 December 2014.
41. Liu, D.; Chen, T.; Liu, S.; Zhou, J.; Zhou, S.; Teman, O.; Feng, X.; Zhou, X.; Chen, Y. PuDianNao: A Polyvalent Machine Learning Accelerator. *SIGARCH Comput. Archit.* **2015**, *43*, 369–381. [CrossRef]
42. Liu, S.; Du, Z.; Tao, J.; Han, D.; Luo, T.; Xie, Y.; Chen, Y.; Chen, T. Cambricon: An Instruction Set Architecture for Neural Networks. In Proceedings of the ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA), Seoul, Korea, 18–22 June 2016.
43. Jouppi, N.P.; Young, C.; Patil, N.; Patterson, D.; Agrawal, G.; Bajwa, R.; Bates, S.; Bhatia, S.; Boden, N.; Borchers, A.; et al. In-datacenter Performance Analysis of a Tensor Processing Unit. In Proceedings of the 2017 ACM/IEEE 44th Annual International Symposium on Computer Architecture, Toronto, ON, Canada, 24–28 June 2017.
44. Betker, M.R.; Fernando, J.S.; Whalen, S.P. The History of the Microprocessor. *Bell Labs Tech. J.* **1997**, *2*, 29–56. [CrossRef]
45. Furber, S. Microprocessors: The Engines of the Digital Age. *Proc. R. Soc.* **2017**, *473*, 20160893. [CrossRef]
46. Brunette, E.S.; Flemmer, R.C.; Flemmer, C.L. A Review of Artificial Intelligence. In Proceedings of the 4th International Conference on Autonomous Robots and Agents, Wellington, New Zealand, 10–12 February 2009.
47. Bush, V. The Differental Analyzer. *J. Frankl. Inst.* **1931**, *212*, 447–488. [CrossRef]
48. Turing, A.M. On Computable Numbers, with an Application to the Entscheidungs problem. *Proc. Lond. Math. Soc.* **1937**, *2*, 230–265. [CrossRef]
49. Strachey, C. Logical or Non-Mathematical Programmes. In Proceedings of the 1952 ACM National Meeting, Toronto, ON, Canada, 1 June 1952; pp. 46–49.
50. Samuel, A. Some Studies in Machine Learning using the Game of Checkers. *IBM J.* **1959**, *3*, 210–299. [CrossRef]
51. Newell, A.; Simon, H.A. The Logic Theory Machine a Complex Information Processing System. 15 June 1956. Available online: http://shelf1.library.cmu.edu/IMLS/MindModels/logictheorymachine.pdf (accessed on 10 April 2021).
52. McCarthy, J. Recursive Functions of Symbolic Expressions and their Computation by Machine, Part I. *Commun. ACM* **1960**, *3*, 184–195. [CrossRef]
53. Minsky, M.; Papert, S.A. *Perceptrons: An introduction to Computational Geometry*; MIT Press: Cambridge, MA, USA, 1969.
54. Nof, S.Y. *Handbook of Industrial Robotics*, 2nd ed.; John Wiley & Sons: New York, NY, USA, 1999; pp. 3–5.
55. Slagle, J.R. A Heuristic Program that Solves Symbolic Integration Problems in Freshman Calculus. *J. ACM* **1963**, *10*, 507–520.
56. Bobrow, D.G. A Question-Answering System for High School Algebra Word Problems. In Proceedings of the Fall Joint Computer Conference, Part I (AFIPS '64 (Fall, Part I)), New York, NY, USA, 27–29 October 1964.
57. Raphael, B. *Robot Research at Stanford Research Institute*; Stanford Research Institute: Menlo Park, CA, USA, 1972.
58. Winograd, T. Procedures as a Representation for Data in a Computer Program for Understanding Natural Language. *Cogn. Psychol.* **1972**, *3*, 1–191. [CrossRef]
59. Pierce, J.R.; Carroll, J.B.; Hamp, E.P.; Hays, D.G.; Hockett, C.F.; Oettinger, A.G.; Perlis, A. *Languages and Machines: Computers in Translation and Linguistics*; The National Academies Press: Washington, DC, USA, 1966. [CrossRef]
60. *Artificial Intelligence: A Paper Symposium*; Science Research Council: London, UK, 1973.
61. Buchanan, B.G.; Feigenbaum, E.A. Dendral and Meta-dendral: Their Applications Dimension. *Artif. Intell.* **1978**, *11*, 5–24. [CrossRef]
62. Colmerauer, A.; Roussel, P. The Birth of Prolog. *Assoc. Comput. Mach.* **1993**, *28*, 37–52.
63. Crevier, D. *AI: The Tumultuous Search for Artificial Intelligence*; Basic Books: New York, NY, USA, 1993; p. 198.

64. Enslow, B. The Payoff from Expert Systems. *Across Board* **1989**, 54. Available online: https://stacks.stanford.edu/file/druid:sb599zp1950/sb599zp1950.pdf (accessed on 6 May 2021).
65. McKinzie, W. The fifth generation. *Proc. IEEE* **1985**, *73*, 493–494. [CrossRef]
66. Hopfield, J.J. Neural Networks and Physical Systems with Emergent Collective Computational Abilities. *Proc. Natl. Acad. Sci. USA* **1982**, *79*, 2554–2558. [CrossRef]
67. Rumelhart, D.E.; Hinton, G.E.; Williams, R.J. Learning Internal Representations by Error Propagation. *Calif. Univ. San Diego La Jolla Inst. Cogn. Sci.* **1986**, *1*, 318–362.
68. Coats, P.K. Why Expert Systems Fail. *Financ. Manag.* **1988**, *17*, 77–86. [CrossRef]
69. Boone, G.W. Variable Function Programmed Calculator. US Patent 4,074,351, 14 February 1978.
70. Laws, D. *Motorola 6800 Oral History Panel: Development and Promotion*; Computer History Museum: Tempe, AZ, USA, 2008.
71. IEEE Standard for Binary Floating-Point Arithmetic. *ANSI/IEEE Stand.* **1985**, *754*, 1–20.
72. Patterson, D. Reduced Instruction Set Computers. *Commun. Assoc. Mach.* **1985**, *28*, 14. [CrossRef]
73. Mead, C.; Conway, L. *Introduction to VSLI Systems*; Addison-Wesley: Boston, MA, USA, 1980.
74. The Street How Cadence Designs the Future. Available online: https://www.thestreet.com/tech/news/cadence072120 (accessed on 7 April 2021).
75. Freeman, R.; Kawa, J.; Singhal, K. Synopsys' Journey to Enable TCAD and EDA Tools for Superconducting Electronics. 2020. Available online: https://www.synopsys.com/content/dam/synopsys/solutions/documents/gomac-synopsys-supertools-paper.pdf (accessed on 7 April 2021).
76. Yilmaz, M.; Erdogan, E.S.; Roberts, M.B. Introduction to Mentor Graphics Design Tools. 2009. Available online: http://people.ee.duke.edu/~{}jmorizio/ece261/LABMANUALS/mentor_toolsv7_windows.pdf (accessed on 7 April 2021).
77. How Intel Makes Chips: Transistors to Transformations. Available online: https://www.intel.com/content/www/us/en/history/museum-transistors-to-transformations-brochure.html (accessed on 7 April 2021).
78. Patterson, D.A.; Sequin, C.H. RISC I: A Reduced Instruction Set VLSI Computer. In Proceedings of the 8th Annual Symposium on Computer Architecture, Washington, DC, USA, 27 June–2 July 1981.
79. Radin, G. The 801 Minicomputer. *IBM J. Res. Dev.* **1982**, *27*, 237–246. [CrossRef]
80. Hennessy, J.; Hennessy, J.N.; Przybylski, S.; Rowen, C.; Gross, T.; Baskett, F.; Gill, J. MIPS: A Microprocessor Architecture. *SIGMICRO Newsl.* **1982**, *13*, 17–22. [CrossRef]
81. MC68030 Product Summary Page-Freescale. 2012. Available online: https://web.archive.org/web/20141006204732/http://www.freescale.com/webapp/sps/site/prod_summary.jsp?code=MC68030 (accessed on 6 April 2021).
82. Intel 80486 DX Microprocessor Datasheet. Available online: https://datasheetspdf.com/pdf-file/493187/Intel/80486DX/1 (accessed on 6 April 2021).
83. Patterson, D.; Hennessy, J. *Computer Organization and Design: The Hardware Software Interface*, 5th ed.; Morgan Kaufmann: Burlington, MA, USA, 2013.
84. Smith, E.; Weiss, S. PowerPC601 and Alpha21064: A Tale of Two RISCs. *Computer* **1994**, *27*, 46–58. [CrossRef]
85. Intel® Pentium® III Processor 1.00 GHz, 256K Cache, 133 MHz FSB. Available online: https://ark.intel.com/content/www/us/en/ark/products/27529/intel-pentium-iii-processor-1-00-ghz-256k-cache-133-mhz-fsb.html (accessed on 6 April 2021).
86. Welker, M.W. *AMD Processor Performance Evaluation Guide*; ADVANCED MICRO DEVICES One AMD Place: Sunnyvale, CA, USA, 2005.
87. Jagger, D. *Advanced RISC Machines Architecture Reference Manual*, 1st ed.; Prentice Hall: Englewood Cliffs, NJ, USA, 1997.
88. Furber, S. *ARM System-on-Chip Architecture*; Addison Wesley: Boston, MA, USA, 2020.
89. Gelsinger, P. Microprocessors for the new millennium: Challenges, opportunities, and new frontiers. In Proceedings of the 2001 IEEE International Solid-State Circuits Conference, Digest of Technical Papers, ISSCC, San Francisco, CA, USA, 7 February 2001.
90. Asanovic, K.; Bodik, R.; Catanzaro, B.C.; Gebis, J.J.; Husbands, P.; Keutzer, K.; Patterson, D.A.; Plishker, W.L.; Shalf, J.; Williams, S.W.; et al. *The Landscape of Parallel Computing Research: A View from Berkeley*; Electrical Engineering and Computer Sciences University of California at Berkeley: Berkeley, CA, USA, 2006.
91. Dual-Core Processors Microprocessors Types and Specifications. 12 June 2006. Available online: https://www.informit.com/articles/article.aspx?p=481859&seqNum=21 (accessed on 6 April 2021).
92. R. Ramanathan Intel® Multi-Core Processors Making the Move to Quad-Core and Beyond White Paper Intel Multi-Core Processors. Available online: https://web.cse.ohio-state.edu/~{}panda.2/775/slides/intel_quad_core_06.pdf (accessed on 6 April 2021).
93. Hong, S.; Graves, L. AMD Announces Worldwide Availability of AMD Ryzen™ PRO 3000 Series Processors Designed to Power the Modern Business PC. 2019. Available online: https://www.amd.com/en/press-releases/2019-09-30-amd-announces-worldwide-availability-amd-ryzen-pro-3000-series-processors (accessed on 2 June 2021).
94. Havemann, R.; Hutchby, J. High-performance interconnects: An integration overview. *Proc. IEEE* **2001**, *89*, 586–601. [CrossRef]
95. Le Cun, Y.; Jackel, L.D.; Boser, B.; Denker, J.S.; Graf, H.P.; Guyon, I.; Henderson, D.; Howard, R.E.; Hubbard, W. Handwritten Digit Recognition: Applications of Neural Network Chips and Automatic Learning. *IEEE Commun. Mag.* **1989**, *27*, 41–46. [CrossRef]
96. Holler, M.; Tam, S.; Castro, H.; Benson, R. An Electrically Trainable Artificial Neural Network (ETANN) with 10240 floating gate synapses. In Proceedings of the International Joint Conference on Neural Networks, Washington, DC, USA, June 1989.

97. Castro, H.A.; Tam, S.M.; Holler, M.A. Implementation and Performance of an Analog Nonvolatile Neural Network. *Analog. Integr. Circuits Signal Process.* **1993**, *4*, 97–113. [CrossRef]
98. Ramacher, U.; Raab, W.; Hachmann, J.U.; Beichter, J.; Bruls, N.; Wesseling, M.; Sicheneder, E.; Glass, J.; Wurz, A.; Manner, R. SYNAPSE-1: A High-Speed General Purpose Parallel Neurocomputer System. In Proceedings of the 9th International Parallel Processing Symposium, Santa Barbara, CA, USA, 25–28 April 1995.
99. Sackinger, E.; Boser, B.E.; Bromley, J.; LeCun, Y.; Jackel, L.D. Application of the ANNA Neural Network Chip to High-Speed Character Recognition. *IEEE Trans. Neural Netw.* **1992**, *3*, 498–505. [CrossRef] [PubMed]
100. Cortes, C.; Vapnik, V. Support-vector networks. *Mach. Learn. Mach. Learn.* **1995**, *20*, 273–297. [CrossRef]
101. Dang, H.; Liang, Y.; Wei, L.; Li, C.; Dang, S. Artificial Neural Network Design for Enabling Relay Selection by Supervised Machine Learning. In Proceedings of the 2018 Eighth International Conference on Instrumentation & Measurement, Computer, Communication and Control (IMCCC), Harbin, China, 19–21 July 2018.
102. Amirhossein, T.; Anthony, S. A Spiking Network that Learns to Extract Spike Signatures from Speech Signals. *Neurocomput* **2017**, *240*, 191–199.
103. Yann, C.J.B.; LeCun, Y.; Cortes, C. The MNIST Database of Handwritten Digits. Available online: http://yann.lecun.com/exdb/mnist/ (accessed on 6 April 2021).
104. Krizhevsky, A.; Nair, V.; Hinton, G. The CIFAR-10 Dataset. Available online: https://www.cs.toronto.edu/~{}kriz/cifar.html (accessed on 6 April 2021).
105. Deng, J.; Dong, W.; Socher, R.; Li, L.J.; Li, K.; Fei-Fei, L. Imagenet: A large-scale Hierarchical Image Database. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009.
106. Deng, L.; Li, J.; Huang, J.T.; Yao, K.; Yu, D.; Seide, F.; Seltzer, M.; Zweig, G.; He, X.; Williams, J.; et al. Recent Advances in Deep Learning for Speech Research at Microsoft. In Proceedings of the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, BC, Canada, 26–31 May 2013; pp. 8604–8608.
107. Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. ImageNet Large Scale Visual Recognition Challenge. *Int. J. Comput.* **2015**, *115*, 211–252. [CrossRef]
108. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet Classification with Deep Convolutional Neural Networks. *Proc. NIPS* **2012**, 1097–1105. [CrossRef]
109. BASU, S. A Cursory Look at Parallel Architectures and Biologically Inspired Computing. In *Academic Press Series in Engineering, Soft Computing and Intelligent Systems*; Academic Press: Cambridge, MA, USA, 2000; pp. 185–216.
110. Johnson, O.; Omosehinmi, D. Comparative Analysis of Single-Core and Multi-Core Systems. *Int. J. Comput. Sci. Inf. Technol.* **2015**, *7*, 117–130. [CrossRef]
111. James, R. Intel AVX-512 Instructions. 2017. Available online: https://software.intel.com/content/www/cn/zh/develop/articles/intel-avx-512-instructions.html (accessed on 3 April 2021).
112. Raskulinec, G.M.; Fiksman, E. SIMD Functions Via OpenMP. In *High Performance Parallelism Pearls*; Morgan Kaufmann: Burlington, MA, USA, 2015.
113. Arm Neon Intrinsics Reference for ACLE Q3. Available online: https://developer.arm.com/architectures/system-architectures/software-standards/acle (accessed on 3 April 2021).
114. Vasudevan, A.; Anderson, A.; Gregg, D. Parallel Multi Channel Convolution using General Matrix Multiplication. In Proceedings of the 2017 IEEE 28th International Conference on Application-specific Systems Architectures and Processors (ASAP), Seattle, WA, USA, 10–12 July 2017.
115. Chellapilla, K.; Puri, S.; Simard, P. High Performance Convolutional Neural Networks for Document Processing. In Proceedings of the Tenth International Workshop on Frontiers in Handwriting Recognition, La Baule, France, 1 October 2006.
116. Intel Math Kernel Library. Available online: https://software.intel.com/en-us/mkl (accessed on 6 April 2021).
117. Vedaldi, A.; Jia, Y.; Shelhamer, E.; Donahue, J.; Karayev, S.; Long, J.; Darrell, T. Convolutional Architecture for Fast Feature Embedding. In Proceedings of the ACM International Conference on Multimedia, MM'14, Orlando, FL, USA, 7 November 2014.
118. Intel Deep Learning Boost (Intel DL Boost). Available online: https://www.intel.com/content/www/us/en/artificial-intelligence/deep-learning-boost.html (accessed on 7 April 2021).
119. Horowitz, M. Computing's Energy Problem (and What We Can Do About It). In Proceedings of the 2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC), San Francisco, CA, USA, 9–13 February 2014.
120. Rodriguez, A.; Segal, E.; Meiri, E.; Fomenko, E.; Kim, Y.; Shen, H.; Ziv, B. Lower Numerical Precision Deep Learning Inference and Training. *Intel White Paper* **2018**, *3*, 1–19.
121. bfloat16-Hardware Numerics Definition. 2018. Available online: https://software.intel.com/content/www/us/en/develop/download/bfloat16-hardware-numerics-definition.html (accessed on 6 April 2021).
122. Developments in the Arm A-Profile Architecture: Armv8.6-A. Available online: https://community.arm.com/developer/ip-products/processors/b/processors-ip-blog/posts/arm-architecture-developments-armv8-6-a (accessed on 7 April 2021).
123. rocBLAS Documentation-Advanced Micro Devices. Available online: https://rocblas.readthedocs.io/en/master/index.html (accessed on 7 April 2021).
124. The Intel®Xeon Phi™ Product Family Product Brief. Available online: https://www.intel.com/content/dam/www/public/us/en/documents/product-briefs/high-performance-xeon-phi-coprocessor-brief.pdf (accessed on 6 April 2021).

125. Capra, M.; Bussolino, B.; Marchisio, A.; Shafique, M.; Masera, G.; Martina, M. An Updated Survey of Efficient Hardware Architectures for Accelerating Deep Convolutional Neural Networks. *Future Internet* **2020**, *12*, 113. [CrossRef]
126. Sermanet, P.; Eigen, D.; Zhang, X.; Mathieu, M.; Fergus, R.; LeCun, Y. OverFeat: Integrated recognition, localization and detection using convolutional networks. *arXiv* **2013**, arXiv:1312.6229.
127. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-scale Image Recognition. *arXiv* **2014**, arXiv:1409.1556.
128. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going Deeper with Convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 1–9.
129. Lin, M.; Chen, Q.; Yan, S. Network in Network. *arXiv* **2013**, arXiv:1312.4400.
130. Szeged, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; Wojna, Z. Rethinking the Inception Architecture for Computer Vision. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 26 June–1 July 2016.
131. Szegedy, C.; Ioffe, S.; Vanhoucke, V.; Alemi, A. Inception-V4, inception-ResNet and the Impact of Residual Connections on Learning. In Proceedings of the AAAI Conference on Artificial Intelligence, San Francisco, CA, USA, 4–9 February 2017; pp. 1–3.
132. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vega, NV, USA, 26 June–1 July 2016; pp. 770–778.
133. Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z.; Lin, Z.; Desmaison, A.; Antiga, L.; Lerer, A. Automatic Differentiation in PyTorch. In Proceedings of the NIPS 2017 Workshop on Autodiff, Long Beach, CA, USA, 9 December 2017.
134. Abadi, M.; Agarwal, A.; Barham, P.; Brevdo, E.; Chen, Z.; Citro, C.; Corrado, G.S.; Davis, A.; Dean, J.; Devin, M.; et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. 2015. Available online: Tensorflow.org (accessed on 4 June 2021).
135. Chetlur, S.; Woolley, C.; Vandermersch, P.; Cohen, J.; Tran, J.; Catanzaro, B.; Shelhamer, E. cuDNN: Efficient Primitives for Deep Learning. *arXiv* **2014**, arXiv:1410.0759.
136. NVIDIA CUDA-X GPU-Accelerated Libraries. 2020. Available online: https://developer.nvidia.com/gpu-accelerated-libraries (accessed on 6 April 2021).
137. Nvidia Tesla V100 GPU Architecture. 2017. Available online: https://images.nvidia.com/content/technologies/volta/pdf/437317-Volta-V100-DS-NV-US-WEB.pdf (accessed on 6 April 2021).
138. Nvidia A100 Tensor core GPU Architecture. 2020. Available online: https://www.nvidia.com/content/dam/en-zz/Solutions/Data-Center/nvidia-ampere-architecture-whitepaper.pdf (accessed on 4 June 2021).
139. Aimar, A.; Mostafa, H.; Calabrese, E.; Rios-Navarro, A.; Tapiador-Morales, R.; Lungu, I.; Milde, M.; Corradi, F.; Linares-Barranco, A.; Liu, S. NullHop: A Flexible Convolutional Neural NetworkAccelerator Based on Sparse Representations of Feature Maps. *IEEE Trans. Neural Netw. Learn. Syst.* **2019**, *30*, 644–656. [CrossRef]
140. Gokhale, V.; Jin, J.; Dundar, A.; Martini, B.; Culurciello, E. A 240 G-ops/s mobile Coprocessor for Deep Neural Networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, Columbus, OH, USA, 23–28 June 2014.
141. Sankaradas, M.; Jakkula, V.; Cadambi, S.; Chakradhar, S.; Durdanovic, I.; Cosatto, E.; Graf, H.P. A Massively Parallel Coprocessor for Convolutional Neural Networks. In Proceedings of the 2009 20th IEEE International Conference on Application-specific Systems, Architectures and Processors, Boston, MA, USA, 7–9 July 2009; pp. 53–60.
142. Du, Z.; Fasthuber, R.; Chen, T.; Ienne, P.; Li, L.; Luo, T.; Feng, X.; Chen, Y.; Temam, O. ShiDianNao: Shifting Vision Processing Closer to the Sensor. In Proceedings of the 42nd Annual International Symposium on Computer Architecture, Portland, OR, USA, 13–17 June 2015; pp. 92–104.
143. Chen, T.; Du, Z.; Sun, N.; Wang, J.; Wu, C.; Chen, Y.; Temam, O. DianNao: A Small-footprint High-throughput Accelerator for Ubiquitous Machine-Learning. *Proc. ASPLOS* **2014**, *4*, 269–284.
144. Chen, Y.; Krishna, T.; Emer, J.; Sze, V. Eyeriss: An Energy-Efficient Reconfigurable Accelerator for Deep Convolutional Neural Networks. *IEEE J. Solid-State Circuits* **2017**, *52*, 127–138. [CrossRef]
145. Google I/O'17 California: Google. Available online: https://events.google.com/io2017/ (accessed on 7 April 2021).
146. Google Cloud Next'18. California: Google. Available online: https://cloud.withgoogle.com/next18/sf/ (accessed on 7 April 2021).
147. Chen, X.; Lin, X. Big Data Deep Learning: Challenges and Perspectives. *IEEE Access* **2014**, *2*, 514–525. [CrossRef]
148. Shabbir, J.; Anwer, T. Artificial Intelligence and its Role in Near Future. *arXiv* **2018**, arXiv:1804.01396.
149. Pavlidis, V.F.; Savidis, I.; Friedman, E.G. *Three-Dimensional Integrated Circuit Design*; Morgan Kaufmann: San Francisco, CA, USA, 2009.
150. Patterson, D.A. Microprocessors in 2020. *Sci. Am.* **1995**, *273*, 62–67.
151. Jouppi, N.P.; Yoon, D.H.; Kurian, G.; Li, S.; Patil, N.; Laudon, J.; Young, C.; Patterson, D. A domain-specific supercomputer for training deep neural networks. *Commun. ACM* **2020**, *63*, 67–78. [CrossRef]
152. Biggio, B.; Fumera, G.; Roli, F. Security Evaluation of Pattern Classifiers under Attack. *IEEE Trans. Knowl. Data Eng.* **2013**, *26*, 984–996. [CrossRef]
153. Finlayson, S.G.; Bowers, J.D.; Ito, J.; Zittrain, J.L.; Beam, A.L.; Kohane, I.S. Adversarial Attacks on Medical Machine Learning. *Science* **2019**, *363*, 1287–1289. [CrossRef] [PubMed]