AI: From Deep Learning to In-Memory Computing

Hsiang-Lan Lung
Macronix, 680 North McCarthy Blvd., Suite 200 Milpitas, CA, USA 95035

ABSTRACT

In the past few years, Artificial Intelligence (AI) has been a subject of intense media hype. Machine learning, deep learning (DL), and AI come up in countless articles, often outside of technology-minded publications. As the AI hype keeps growing, it is important to be able to recognize the signal in the noise, to tell apart world-changing developments from what are merely over-hyped press releases. This paper tries to explain how deep learning is working and how GPU (Graphic Processing Unite) can make it a reality. Finally, in-memory computing (IMC) for DL is introduced to point out future high performance and low power DL hardware development direction.

Keywords: AI, machine learning, deep learning, artificial neuron network, ANN, convolution neuron network, CNN, in-memory computing, computing in memory

1. INTRODUCTION

During recent years, the DL [1] has increasingly improved on tasks such as classifying images, understanding speech, playing video games and translating between languages to a nearly human skill level. More importantly, these developments have allowed DL systems to become commercially widespread in influencing social media sites, shopping and recommender systems, banking and finance, numerous cloud-based computing applications, and even mobile phones and living rooms. The DL model is like a huge self-organized trial-and-error machine with millions of tunable parameters. After feeding the machine with big data and proceeding to iterate for tens or hundreds of millions of training cycles, the machine can find the best parameters and weights related to the DL model. Currently, GPU cards are the best hardware solution for DL, because of their excellent in-parallel matrix multiplication capability and supported software. However, their flexibility (gaming support) makes them less efficient for DL, and this is where other DL accelerators ASIC (Application-Specific Integrated Circuit) come in to provide better efficiency and performance. But both the GPU and ASIC are built on the traditional Von-Neumann architecture. The time and energy spent transporting data between memory and processor (the so-called “Von-Neumann bottleneck") has become problematic, especially for data-centric applications such as real-time image recognition and natural language processing. In order to achieve even larger acceleration factors and lower power beyond the Von-Neumann architecture, IMC based on non-volatile memory (NVM) array, such as phase change memory (PCM) and resistive random access memory (RRAM) has been explored. The IMC’s vector-matrix multiplication replaces the expensive high power consumed matrix multiplication operation in CPU/GPU (digital circuits) and avoids moving weights from/to memory. As a result, it has great potential to have a huge impact on the performance and power consumption of DL.

2. A BRIEF HISTORY OF AI

AI was born in the 1950s [2], as a handful of pioneers from the just beginning field of computer science started asking if computers could be made to "think"—a question whose consequence we are still exploring today. A concise definition of the field would be the effort to automate intellectual tasks normally performed by humans. For a fairly long time many experts believed that human-level artificial intelligence could be achieved simply by having programmers handcraft a sufficiently large set of clear rules for manipulating knowledge. Figure 1 shows the three booms of AI. The first boom is GOFAI. John Haugeland gave the name GOFAI ("Good Old-Fashioned Artificial Intelligence") [3]. It is the term for the collection of all methods in artificial intelligence research that are based on high-level "symbolic" (human-readable) representations of problems, logic and search. A major part of GOFAI is heuristic search, a function that ranks alternatives in search algorithms at each branching step based on available information to decide which branch to follow. The second boom is the expert system, which is a computer system that contains large amounts of knowledge, rules and reasoning mechanisms to provide solutions to real-world problems to emulate the decision-making ability of a human expert. Expert systems are designed to solve complex problems by reasoning through bodies of knowledge, represented mainly as if–then rules rather than through conventional procedural code. The first expert systems were created in the 1970s and then expanded in the 1980s [4]. Although heuristic search and expert system proved suitable to solve well-

Metrology, Inspection, and Process Control for Microlithography XXXIII, edited by
Vladimir A. Ukraintsev, Ofer Adan, Proc. of SPIE Vol. 10959, 109591J © 2019
SPIE · CCC code: 0277-786X/19/$18 · doi: 10.1117/12.2517237

defined, logical problems, such as playing chess, it turned out to be intractable to figure out explicit rules for solving more complex, fuzzy problems, such as image classification, speech recognition, or language translation. The third boom of AI arose to take their place: machine learning (ML).

ML is not a new concept. In 1959, Arthur Samuel, one of the pioneers of ML, defined machine learning as a "field of study that gives computers the ability to learn without being explicitly programmed" [5]. That is, ML programs have not been clear entered into a computer, like the if-then statements above. ML programs, in a sense, adjust themselves in response to the data they're exposed to. ML is a sub branch of AI that focuses on teaching computers how to learn without the need to be programmed for specific tasks [6]. In fact, the key idea behind ML is that it is possible to create algorithms that learn from and make predictions on data. It has the ability to modify itself when exposed to more data; i.e. machine learning is dynamic and does not require human intervention to make certain changes. That makes it less brittle, and less reliant on human experts. ML had waited for 50 years for the development of deep learning algorithms. Deep learning has allowed the ML boom to become a reality alongside the cross point of computer technology (GPU), modern machine learning algorithms (DNN) and the availability of big data. Deep learning is a subset of machine learning, and it is the core of current machine learning as shown in figure 1(B).

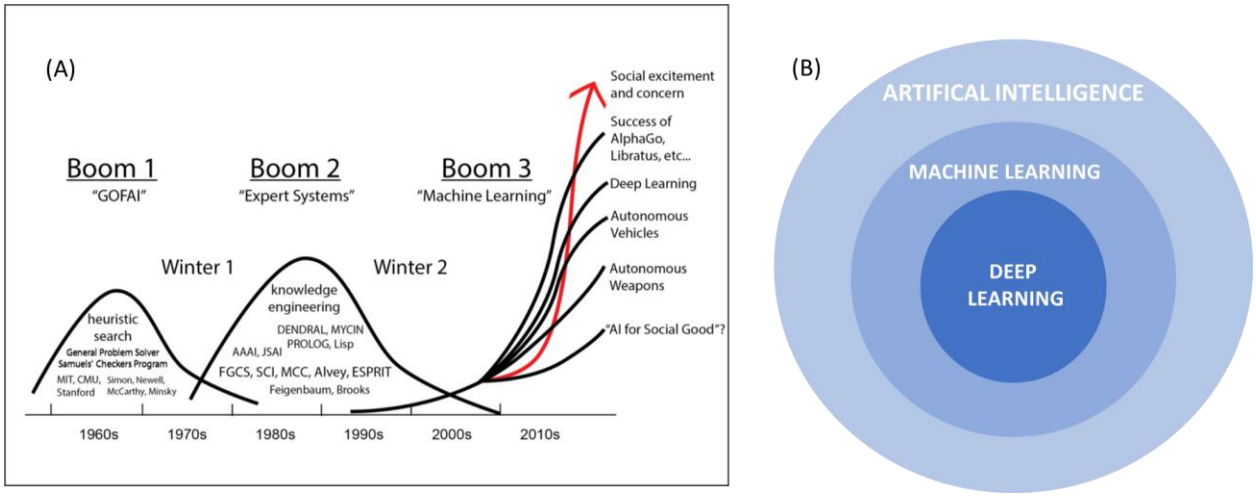

Figure 1. (A) The three booms of AI: GOFAI ("Good Old-Fashioned Artificial Intelligence") symbolic AI, expert system and machine learning [3]. (B) The relation between AI, machine learning and deep learning.

3. ARTIFICIAL NEURAL NETWORK

Artificial Neural Network (ANN) is fundamental framework of DL, and is the combination of artificial neurons. Figure 2 shows an example human neurons and artificial neurons (AN). Similar to a human neuron, AN (Fig. 2B) obtains input information from the dendrites (xi) and gives the output information through an axon (yi). The arrow in Fig.2B indicates the weights of certain inputs or outputs that are similar to the synapse of a neuron. Synapses connect each neuron to form a neural network, and weights connect each node to form an ANN as seen in Fig.2C, 2D. Figure 3 explains how an ANN is trained to predict the value of a house. Predicting (inferencing) the house value is to add up all the product of house's feature and its optimized weighting factor, as shown in the figure 3 (A), as a matrix(vector) inter product of [weights] and [features]. For training, house's feature data is fed into the ANN to get the predicted house value with the previous trained weights combination. The difference between the predicted value and the real value is the loss (error). The goal of training is to modify the weight combinations to minimize the loss. How does one minimize loss? One way is to build a framework that multiplies inputs in order to make guesses to the input nature. Different outputs/guesses are the product of the inputs and the algorithm. Usually, the initial guesses are quite wrong, and if one is lucky enough to have ground-truth labels pertaining to the input, one can measure how wrong their guesses are by contrasting them with the truth, and then use that error to modify the algorithm. That is what neural networks do. They keep on measuring the loss (error) and modifying their parameters until they can't achieve any less error. They are, in short, an optimization algorithm. If tuned right, they minimize their error by guessing and guessing and guessing again.

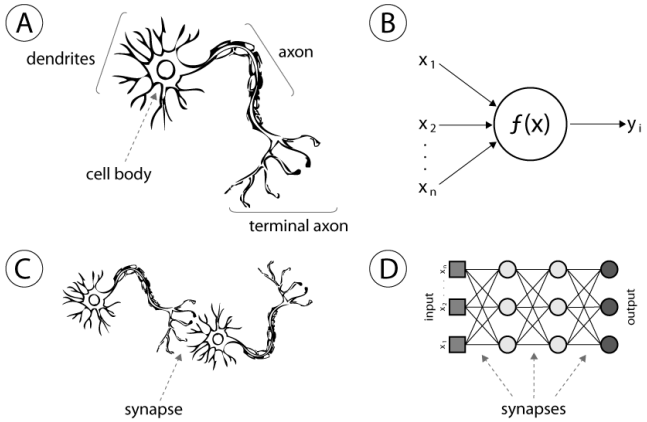

Figure 2. (A) Human neuron (B) artificial neuron or hidden unity (C) biological synapse (D) ANN synapses [7].

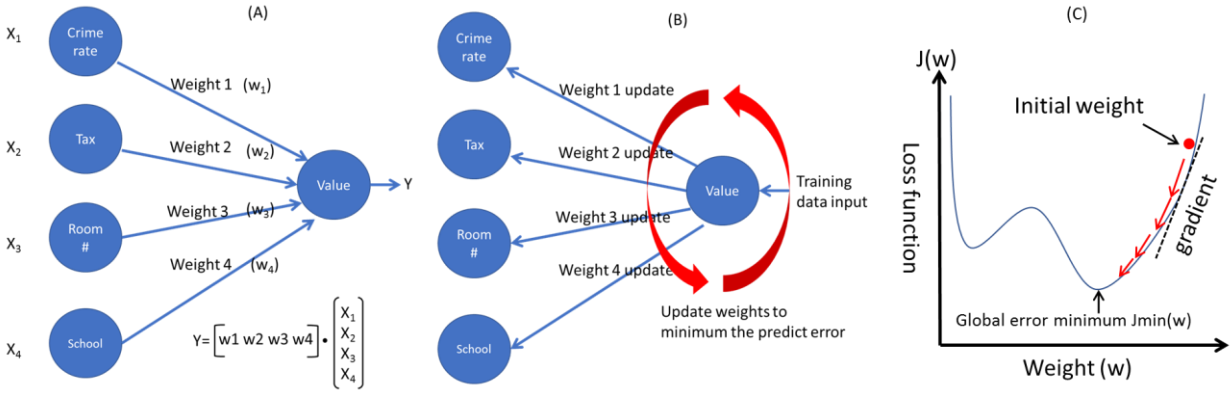

Figure 3. (A) Use an ANN (2 layers, 5 neurons) to predict the value of a house. Crime rate, tax, room number and school district are the features (input X) of the house. The value (output Y) is decided by the weighting (w) process of each feature. (B) Training the ANN (weights) to predict a house value is a loop of the following operations: 1. training data (thousands of real house vale data with different features) is used to calculate the predicted loss. 2. Weights are updated to reduce the loss. (C) To achieve the most accurate ANN is to minimize prediction loss by optimizing the weights combination.

4. DEEP LEARNING

Usually, when people use the term DL, they are referring to deep artificial neural networks (DANN or DNN). Deeper means more layers of ANs are added into the ANN as shown is figure 2 (D). From current trend, the deeper DNN has better preformation. It has set new records in accuracy for many important problems, such as image recognition, sound recognition, recommender systems, etc. For example, deep learning is part of DeepMind's well-known AlphaGo algorithm, which beat the former world champion Lee Sedol at Go in early 2016. But the cost of adding more layers is that it makes the DNN more difficult to train due to more features and weights needing to be optimized. To solve this issue, several functions (layers) – like activation and dropout [8] – are added to the DNN. The activation function is used to introduce non-linear properties to the DNN, and one of the most popular one is ReLU (Rectified Liner Unit) [9] which screens out the negative number. Dropout is to randomly remove certain percentage of neuron (node) in DNN. Convolution neural network (CNN) [10] is one of the most important ANNs that is used to classify or detect objects in computer vision. Figure 4 on the left shows a Python language code of a CNN and on the right is its model to make an image classification of cat, dog and rabbit 3 categories. Essentially, a CNN has two parts. One is the convolution layers for features extraction, and another is the fully connected layers for object classification. The convolution layer starts with a filter (kernel), which is simply a small matrix of weights (first layer has 32 3x3 filters). This filter "slides" over the input data (image represented by pixels/numbers), performing an elementwise multiplication with the part of the input it is currently on, and then summing up the results into a single output pixel. After sided through the whole image, each filter will generate a filtered image as an input of the ReLU operation. Pooling is a process to compress the filtered image. It gathers neighbor pixels to generate a new pixel to produce a compressed filtered image. MaxPooling2D((2,2))

means that only the maximum pixel (number) among the 4 neighboring pixels will be used. The convolutional and pooling layers in CNN are directly inspired by the classic notions of simple cells and complex cells in visual neuroscience [11]. After three convolution operations, the convoluted image will be "flattened" from a 2D array to a 1D vector of pixels (features/neurons) to connect to the fully connected layers. For classification, the operation is similar to figure 3, but the DNN becomes 3 layers. The last hidden layer's 512 nodes (neurons) are connected to next layer's 3 nodes with 512x3=1536 weights (synapse). The values of the 3 output nodes are calculated by vector (512 features)-matrix (3 x 512 weights) multiplication. The largest value in the output node indicates the maximum possibility of the certain object (cat). The batch gradient descent algorithm [12] is used to train this model with RMSprop optimizer [13]. Gradient descent and it varieties are the only algorithm for training a DNN, and its different optimizers are used to improve the training speed. The training process is to adjust the weights in the network layer by layer. The update of each weight (w_{ji}) is: $w_{ji} \leftarrow w_{ji} + \eta \bullet \delta j \bullet x_i$ where η is the learning rate and δ_j is the loss (error) back propagated from the node j in the next neighbor layer. x_i is the input of the node i.

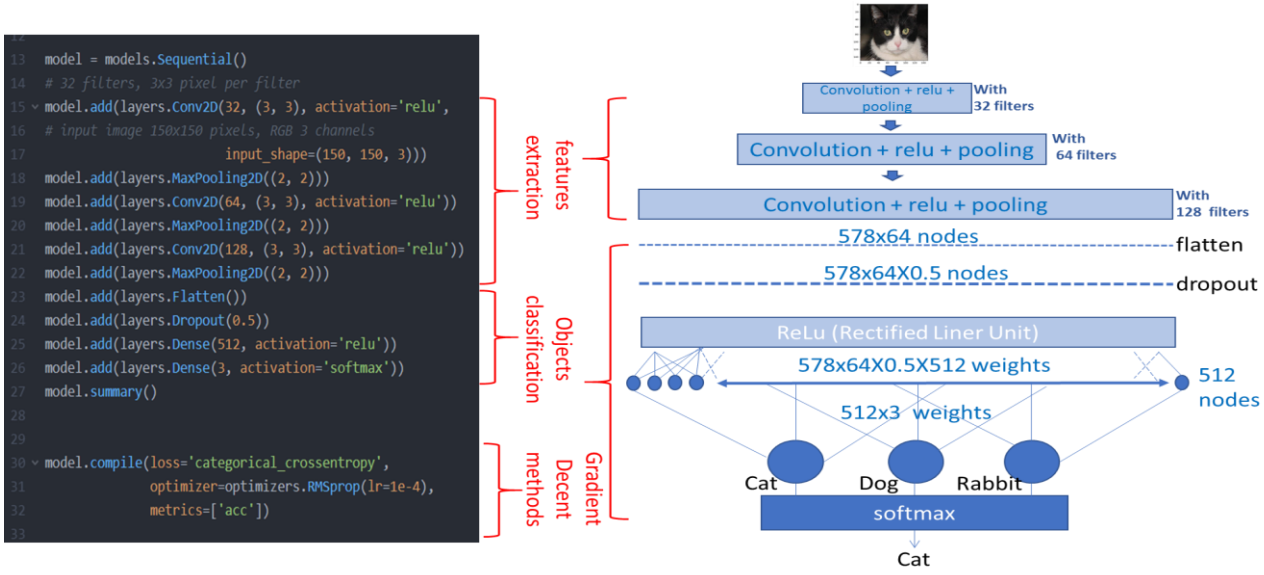

Figure 4. A Python code of a CNN (left) and its mode (right) for objects (cat, dog, and rabbit) classification.

5. HARDWARES FOR DEEP LEARNING

Figure 5 (A) [14] shows DL algorithms are comprised of a spectrum of operations. Although matrix multiplication (gemm) is dominant, optimizing performance efficiency while maintaining accuracy requires the core architecture to efficiently support all of the auxiliary functions. Figure 5 (B) shows the comparison of CPU and GPU. A central processing unit (CPU) is designed to handle complex tasks, such as time slicing, complex control flows and branching, security, etc. In contrast, graphical processing unites (GPUs) only do one thing well. They handle billions of repetitive low level tasks, such as matrix multiplication. GPUs have thousands of arithmetic logic units (ALUs) compared to traditional CPUs that commonly have only 4 or 8. But, the GPU is still a general purpose processor that has to support millions of different applications and software. For every single calculation in the thousands of ALUs, a GPU needs to access registers or shared memory to read and store the intermediate calculation results. Because the GPU performs more parallel calculations on its thousands of ALUs, it also spends proportionally more energy accessing memory and also increases footprint of GPU for complex wiring. To address those issues, ASIC for DL is needed, and TPU (Tensor Processing Unit) [15] is one of the example. Figure 5 (c) is the block diagram of a TPU chip. It is a matrix processor specialized for neural network workloads that can handle the massive multiplications and additions for neural networks, at blazingly fast speeds while consuming much less power and inside a smaller physical footprint. Its key enabler is a major reduction of the von Neumann bottleneck (moving data from memory). By knowing what DNN it's going after, TPU place thousands of multipliers and adders and connect them to each other directly to form a large physical matrix of those operators. For operation, at first, TPU loads the weights from memory into the matrix of multipliers and adders. Then, the TPU loads data (features) from memory. As each multiplication is executed, the result will be passed to next multipliers while taking summation at the same time. So the output will be the summation of all multiplication result

between data and parameters. During the whole process of massive calculations and data passing, no memory access is required at all. TPU's drawback is losing of flexibility; it only supports a few specific neural networks.

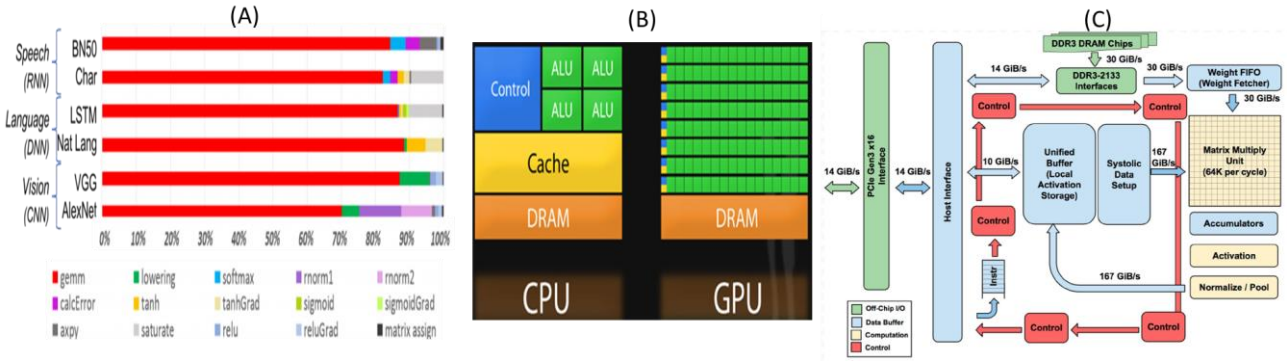

Figure 5. (A) Operations spectrum of deep learning algorithms. (B) Comparison of CPU and GPU. (C) The block diagram of a TPU.

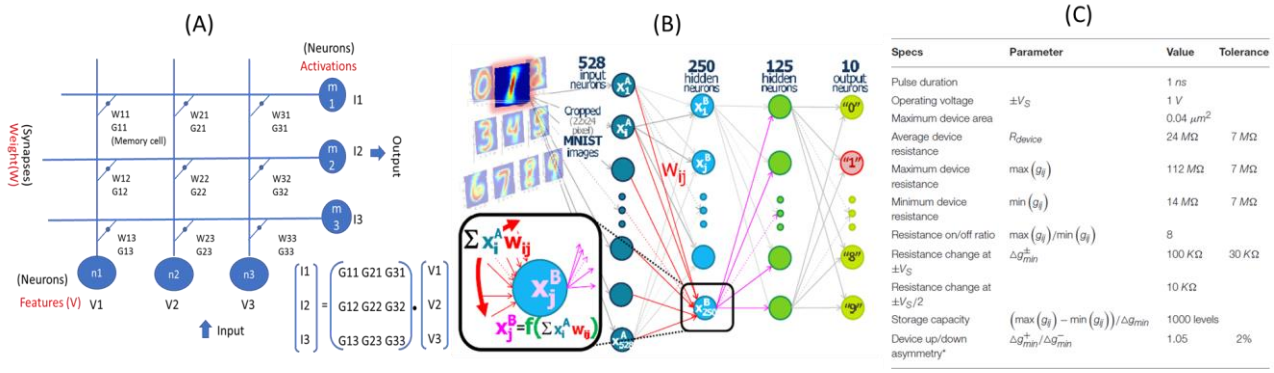

Figure 6. (A) Operations spectrum of deep learning algorithms. (B) Comparison of CPU and GPU. (C) The block diagram of a TPU.

6. IN-MEMORY COMPUTING FOR DEEP LEARNING

The inference and training algorithm of DNN mainly involves computing vector-matrix multiplication in forward and backward directions. This operation can be performed by IMC on a 2D crossbar memory array as it was proposed more than 50 years ago [16]. As shown in figure 6 (A), the weights (G) of the DNN are stored in the memory cells which can be 1T (Transistor)-1R (Resistor) or 1T cell. By applying voltage inputs V at the rows simultaneously and read current outputs I from the columns, analog weighted (G) summation is achieved through Kirchhoff's Current Law and Ohm's Law. In an ideal crossbar memory array, input-output relationship can be represented as: I = V G. The vector-matrix multiplication is performed by mapping input vector to input voltage V, matrix to conductance G, and output to current I. The IMC vector-matrix multiplication replaces the expensive high power consumed matrix multiplication operation in GPU/TPU (digital circuits) and avoids moving weights from memory, and as a result, it greatly improves the Although performance and power consumption of a DNN. Demonstrations for acceleration of DNN training using backpropagation algorithm have reported acceleration factors (vs. CPU) ranging from 27× [17] to 2140× [18] and significant reduction in power and area. Figure 6 (B) shows a demonstration of a DNN inferencing by using PCM devices as one synapse (weight) and each layer's neurons drive the next layer through weights wij and a nonlinearity f(). Input neurons are driven by pixels from successive MNIST images; the 10 output neurons identify which digit was presented [19]. A limitation of IMC DNN acceleration is the imperfection of the memory device. Device characteristics usually considered beneficial for memory applications such as high on/off ratio, digital bit-wise storage, or irrelevant characteristics like asymmetrical set and reset operations are becoming limitations for the acceleration of DNN training. A perfect memory cell for IMC DNN as shown in figure 6 (c) plus the design of a system and CMOS circuitry that imposes specific requirements for the perfect resistive devices, a 30000X acceleration factor can be achieved [20]. the IMC for DNN has great benefits, it still doesn't have products in the market yet. The challenges that prevent it from happening include: 1.The imperfections of the memory cell (cycling endurance, small dynamic range, resistance drift,

asymmetry program). 2. Data transfer between layers (AD, DA conversion, digital function connection). 3. Flexible software, framework support (software reconfigurable IMC DNN).

7. CONCLUSION

With the help of advanced algorithms and computing hardware (GPU), deep learning pushes AI to the next level. Thousands of in-parallel processing ALUs make GPUs powerful machines to carry out matrix multiplication for DNN operation. By sacrificing flexibility, ASIC constructed DNN acceleration chips like TPUs can achieve even higher performance and lower power. But using digital circuits to do matrix multiplication has its own limitations. In order to achieve an even higher acceleration factor and low power, in-memory computation for vector-matrix multiplication of DNN was proposed. Although IMC creates huge benefits for DNN, many challenges still exist. For example: memory cell imperfections, data transfer between layers and supported software and frameworks need to be overcome before IMC DNN becomes reality.

REFERENCES

[1] Yann LeCun, Yoshua Bengio & Geoffrey Hinton, "Deep learning", N AT U R E, Vol. 512, 436-444, (2 0 1 5).
[2] S. Russell & P. Norvig, "Artificial Intelligence: A Modern Approach", Prentice Hall, (2003).
[3] Colin Garvey, "Broken Promises & Empty Threats: The Evolution of AI in the USA, 1956-1996", Society for the history of technology, 12 March, 2018 http://www.technologystories.org/ai-evolution/ (12 January 2019).
[4] John Haugeland, "Artificial Intelligence: The Very Idea", Cambridge, Massachusetts: MIT Press (1985).
[5] Arthur Samuel, "Some Studies in Machine Learning Using the Game of Checkers", IBM Journal of Research and Development. Vol. 3, No.3. July, 535-554 (1959).
[6] C. M. Bishop, "Pattern Recognition and Machine Learning", Springer, (2006).
[7] Vinícius Maltarollo et al, "Applications of Artificial Neural Networks in Chemical Problems", Artificial Neural Networks-Architectures and Applications, ed. K. Suzuki, InTech, 203–223 (2013)
[8] Alex Krizhevsky, Ilya Sutskever & Geoffrey Hinton, "ImageNet Classification with Deep Convolutional Neural Networks", Proc. Advances in Neural Information Processing Systems 25 1090–1098 (2012).
[9] Xavier Glorot, Antoine Bordes & Yoshua Bengio," Deep sparse rectifier neural networks", Proc. 14th International Conference on Artificial Intelligence and Statistics 315–323 (2011).
[10] Yann LeCun, et al., "Handwritten digit recognition with a back-propagation network", Proc. Advances in Neural Information Processing Systems 396–404 (1990).
[11] D. H. Hubel & T. N. Wiesel, "Receptive fields, binocular interaction, and functional architecture in the cat's visual cortex", J. Physiol. 160, 106–154 (1962).
[12] Aerin Kim "Difference between Batch Gradient Descent and Stochastic Gradient Descent", Toward Data Scientist, Sep 17, 2017, https://towardsdatascience.com/a-look-at-gradient-descent-and-rmsprop-optimizers-f77d483ef08b, (18 Jan 2019).
[13] Rohith Gandhi "A Look at Gradient Descent and RMSprop Optimizers", Toward Data Scientist, Jun 19, 2018, https://towardsdatascience.com/a-look-at-gradient-descent-and-rmsprop-optimizers-f77d483ef08b, (18 Jan 2019).
[14] B. Fleischer et al., "A Scalable Multi-Tera OPS Deep Learning Processor Core for AI Training and Inference", Symposium on VLSI Circuits, C4-2, (2018).
[15] Norman P. Jouppi et al., "In-Datacenter Performance Analysis of a Tensor Processing Unit", Proc. of the 44th Annual International Symposium on Computer Architecture, 1-12 (2017).
[16] Kybernetik Steinbuch, "Dielernmatrix", Kybernetik 1, 36–45 (1961).
[17] G.W.Burr et al., "Large-scale neural networks implemented with non-volatile memory as the synaptic weight element: comparative performance analysis (accuracy, speed, and power)", Proc. IEDM 15, 76-79 (2015).
[18] J. Seo et al., "On-chip sparse learning acceleration with CMOS and resistive synaptic devices", IEEE Transactions on Nanotechnology (TNANO), Vol. 14, No. 6, 969-979 (2015).
[19] G.W.Burr et al., "Experimental demonstration and tolerancing of a large-scale neural network (165,000 synapses), using phase-change memory as the synaptic weight element", Proc. IEDM 14, 697-700 (2014).
[20] Tayfun Gokmen & Yurii Vlasov, "Acceleration of deep neural network training with resistive cross-point devices", Front. Neurosci, Volum 10, article 333, (2016).