Chapter 12
Efficient Neuromorphic Systems and Emerging Technologies: Prospects and Perspectives

Abhronil Sengupta, Aayush Ankit, and Kaushik Roy

12.1 Introduction

Deep Learning Networks (DLN) are inspired from the hierarchical organization of neurons and synapses in the human brain and are an important class of machine learning algorithms. Since its inception, it has been widely adopted in multifarious recognition tasks. Lately, DLNs have redefined the state of the art for many cognitive applications including computer vision [1], speech recognition [2], and natural language processing [2] across various application domains. For instance, Baidu's Deep Speech 2 recently demonstrated English and Mandarin language recognition at par to human capabilities. Google's DeepMind recently defeated AlphaGo world champion. Tesla is using deep learning powered vision, sonar and radar processing in their self-driving systems. Also, deep learning is being adopted for ever-more tasks, for example, face recognition by Facebook, data analytics by IBM, recommender systems by Amazon, and so on.

DLN performance (accuracy) is a strong function of the network scale. Hence, the network size has been commensurate with the target accuracy and the problem complexity. LeCun et al. used Convolutional Neural Networks (CNN) for handwritten digits classification using 1 million parameters in 1998 [3]. In 2012, Krizhevsky et al. used a CNN with 60 million parameters for object classification having 1000 classes [4]. Deepface used 120 million parameters to classify human faces [5]. In a nutshell, large DLN models are preferred by Machine Learning practitioners and this necessitates developing efficient systems to power DLNs.

Lately, conventional computing systems which are primarily based on von-Neumann computing model have been extensively used in cognitive applications

A. Sengupta • A. Ankit • K. Roy (✉)
School of Electrical and Computer Engineering, Purdue University, 465 Northwestern Ave, West Lafayette, IN 47907, USA
e-mail: asengup@purdue.edu; aankit@purdue.edu; kaushik@purdue.edu

A. Chattopadhyay et al. (eds.), *Emerging Technology and Architecture for Big-data Analytics*, DOI 10.1007/978-3-319-54840-1_12

and have shown fascinating results across many domains. Majority of the current DLN implementations are done on Graphic Processing Units (GPUs) owing to their ability to deal with intensive parallel computations. For example, Tesla's DLNs uses in-vehicle GPU based supercomputers. Facebook is using GPUs to power its machine learning algorithms. It seems appealing that systems based on conventional computing styles like GPUs are addressing the computational needs of DLNs and hence seems to be an ideal choice for accelerating DLN for cognitive applications. The observation and the inference are correct but this simple treatment of facts veil the skyrocketed power and memory requirements driving the modern computing systems which is many orders higher than human brain.

There has been recent research to work around the memory bottlenecks in GPUs by using techniques such as data batching and virtualized DLNs [6]. However, the latency requirements can be unsuitable for real-time applications. Previous works have also used specialized hardware for DLN acceleration [7]. This work underscores the memory access associated power requirements and shows that the DRAM accesses account for significant power consumption. Consequently, the DLN performance on conventional computing systems is presently limited by the memory and power bottlenecks. Hence, with complex, bigger, and deeper networks that can achieve brain-like cognitive capabilities being developed, there is a dire need to develop brain-like energy-efficient architectures that can drive them.

These power and memory bottlenecks in the conventional computing systems are primarily a consequence of the mismatch between the conventional computing systems and computing patterns involved in these cognitive applications. MOS transistors, being on/off switches, have served as an ideal match to the abstractions of switching functions and Boolean logic, which (together with von Neumann architecture) form the underpinnings of modern computing. While current computing platforms are well suited to applications that involve arithmetic computations and storing and retrieving large amounts of data, they are known to be highly inefficient—requiring orders of magnitude more energy consumption—for performing tasks that humans routinely perform, such as visual recognition, semantic analysis, and reasoning. This inefficiency stems from the realization of neuron and synapse functionality by translating the underlying mathematical functions to Boolean logic gates and subsequently to transistors, resulting in hundreds of transistors to mimic a single neuron/synapses. In this chapter we will review typical von-Neumann computing based CMOS architectures used for DLN implementations and demonstrate the manner in which spintronic crossbar array based "in-memory" computing platforms can lead to more compact and energy-efficient implementations.

12.2 Neural Network Basics

DLNs are feed-forward networks where the computational units—the neurons are connected to neurons in other layers through programmable connections termed as synapses. Figure 12.1a shows an NN with one hidden layer. DLNs are usually

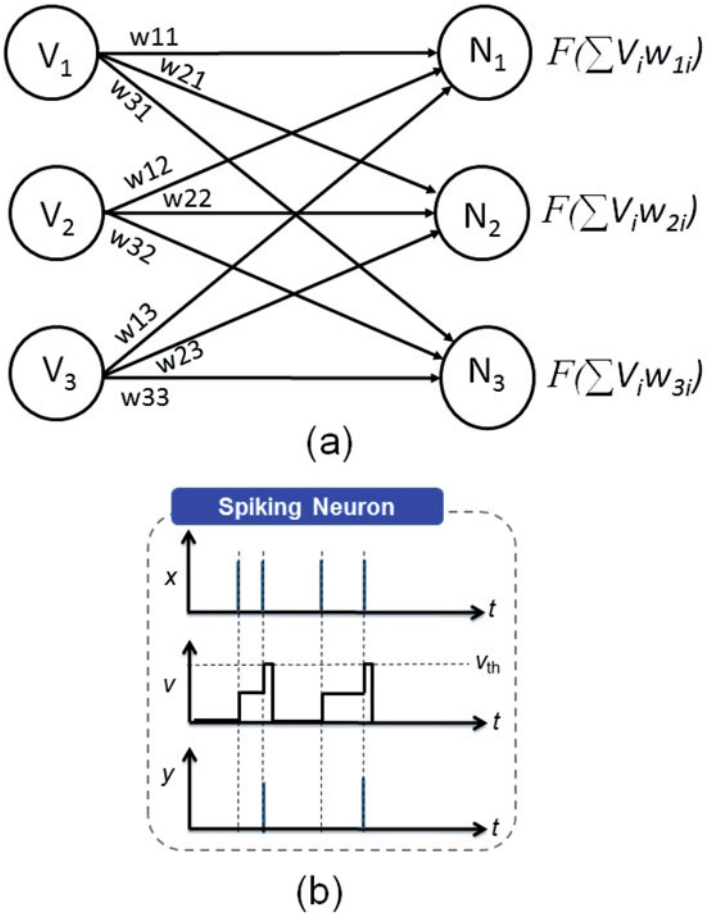

Fig. 12.1 (**a**) Feedforward NN consists of neurons in one layer connected to another layer through programmable synaptic weights, (**b**) IF spiking neuron computing model: x is the input spike train and v is the neuron membrane potential. The neuron generates an output spike y whenever v crosses the threshold voltage v_{th}

characterized by multiple such hidden layers. Depending on the type of connectivity, a DLN can either be fully connected (Multi-Layer Perceptron) or have sparse connectivity (Convolutional Neural Network). The net input a neuron receives is the weighted summation of its inputs. Thus, a neuron computation involves a weight fetch from a memory (SRAM) followed by a multiply-and-accumulate (MAC) operation for every input it receives. Once, all the inputs have been processed for the above-mentioned operations, a non-linear computation is done to compute the neuron's output activation potential. This output then acts as the input for the next layer neurons.

DLN architectures discussed above have achieved record-breaking results for many classification problems but the substantial computational cost of training and running deep networks have motivated the research for the more biologically plausible NN version called Spiking Neural Networks (SNN). In recent years, deep SNNs have become an increasingly active field of research which is primarily motivated by the extremely energy-efficient cognitive processing in human brain. Driven by brain-like event-driven computations, SNNs involve data processing in an input-triggered fashion. Unlike conventional ANNs where a vector is given at the input layer once and the corresponding output is produced after processing through several layers of the network, SNNs require the input to be encoded as a spike train. At a particular instant, each spike is propagated through the layers of the network while the neurons accumulate the spikes over time causing the output neuron to fire or spike. Thus, the spike information is used to communicate between the layers of the network. Figure 12.1b describes the functionality of an Integrate-Fire (IF) spiking neuron. Note that more bio-plausible models also include a leak-term in the membrane potential which causes it to decay in the absence of spikes.

12.3 General Purpose Computing Architecture

In this section we describe a general purpose architecture to implement SNNs. It involves an SRAM which stores the SNN weights and inputs and a computation core to perform the neuronal computations. Figure 12.2 shows the block diagram of the architecture and the logical dataflow between the constituent blocks. The SRAM memory stores the input data (image pixel values and weights) for the trained SNN. Efficient data movement is achieved by buffering the input data—image data and weight data in FIFOs. Image and weight data are read from SRAM memory and processed by an array of Neuron Units (NUs). The NUs keep accumulating weights depending on the input spikes, until all the inputs for a particular neuron are processed. After this, the NUs produce the output spikes which are written back to the SRAM.

Here we discuss the logical dataflow involved between the different components for a time-step in SNN computation. Input data read from the SRAM is stored into the Input FIFO to stream across the NU array as all the neurons in a layer share the same inputs. Corresponding to the input data, weight data are fetched from the SRAM and stored into the weight FIFOs for temporal reuse by the NUs. Each NU receives the weights from its dedicated weight FIFO. The input FIFO is flushed and new set of data are read from SRAM and put in the Input FIFO, after the input processing is finished. Similarly, the weight FIFO gets flushed, new weights read from SRAM and stored into weight FIFO, once the weight processing finishes. When all the computations for the neurons currently scheduled into the NUs are done, the next set of neurons are scheduled into the NUs, corresponding weights read from SRAM into their respective weight FIFOs. Each NU performs "accumulate-and-fire" operation. The NUs are connected in a serial fashion to allow data streaming from Input FIFO to the rightmost NU.

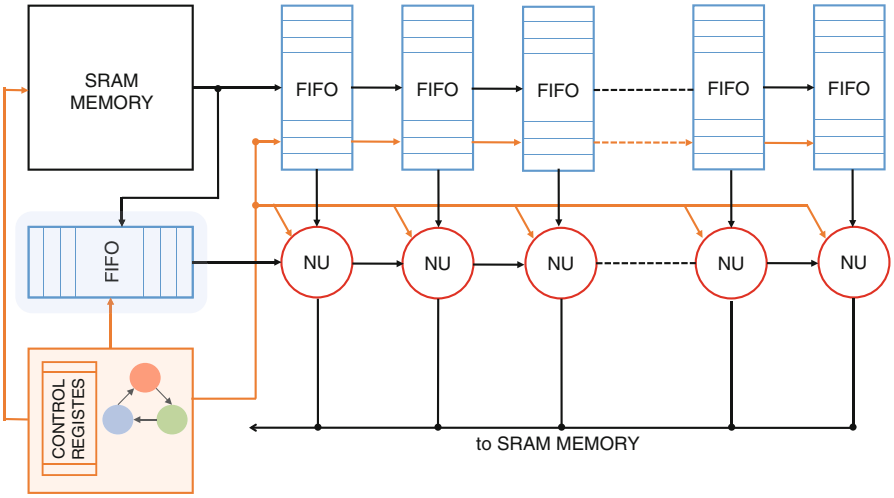

Fig. 12.2 General purpose computing architecture—logical dataflow and the constituent blocks

The neuron computations are done layer wise—read the inputs and weights from SRAM, compute all the outputs corresponding to the first layer, store back the outputs in SRAM and proceed to the next layer. Within a layer neurons are temporally scheduled in the NUs. First, the output computations for the first set of 'N' neurons are done, then the next set of 'N' neurons from the same layer are scheduled in the NU and this goes on until all the neurons in the current layer have been evaluated. Hence, we temporally map different layers of the neural network and different neurons within a layer to compute the entire neural network for a given input data.

For a typical fully connected network, the memory component of energy consumption (access + leakage) on this architecture is more than 50% on an average across various computing workloads. As discussed before, the general purpose computing frameworks have separate processing (core) and data storage components (SRAM). The volume of data to process has drastically increased with increasing DLN size. The DLN size has been ever-increasing to get better accuracy and solve more complex recognition tasks. Consequently, data movement between core and memory has been increasing and is becoming one of the most critical performance and energy bottlenecks in these computing systems. On one hand, this limits the research community to use sub-optimal architectures for solving the recognition problems and on the other hand this impedes their deployment on mobile computing platforms which are energy limited.

The size of the SRAM needed scales directly with the network size and the network size is a strong function of target accuracy and problem complexity. AlexNet [4], for instance, contains 5 convolutional layers with 2 fully connected layers and uses 2.3 million weights. The more recent VGG-16 [1] contains 16 convolution layers and 3 fully connected layers and uses 14.7 million weights. However, larger

memories lead to increased power consumption due to increased memory access and leakage power and the memory component of energy (access + leakage) is typically higher than the computation component. Eventually, the memory-driven limitations affect the size of DLN that can be used. To work around these bottlenecks, Machine Learning practitioners must use less desirable DLN architectures (e.g., smaller number of layers and neurons). Additionally, memory imposed bandwidth also limits the number of NUs one can have in the NU-array for neuronal computations. Hence it is imperative to develop innovative architectures which addresses the rigid memory limitations of general purpose frameworks used towards DLN acceleration.

12.4 Underlying Device Physics

Before we present "In-Memory" computing architectures based on spintronic technologies that can potentially alleviate the memory-bandwidth limitations of conventional CMOS based neuromorphic architectures, let us discuss the underlying device physics of such emerging technologies that can provide a direct mapping to synaptic and neural functionalities.

Let us first illustrate the device structure and principle of operation of a Magnetic Tunnel Junction (MTJ: Fig. 12.3a) [8]. The MTJ consists of two ferromagnets (FMs) separated by a tunneling oxide barrier (MgO). The magnetization direction of one of the layers (denoted by "pinned" layer, PL) is magnetically hardened so that it serves as the reference layer. The magnetization of the "free" layer (FL), can be manipulated by an input charge current. The MTJ is characterized by two stable resistance states, namely the low-resistance parallel (P) configuration ("free" and "pinned" layer magnetizations are parallel) and the high-resistance anti-parallel (AP) configuration ("free" and "pinned" layer magnetizations are anti-parallel). The MTJ can be switched between the two stable states by charge current flow through the stack due to spin-transfer torque (STT) effect generated by charge current flowing through the "pinned" layer [9]. Recent experiments have shown that such an MTJ structure with in-plane magnetic anisotropy (IMA) can also be switched by a charge current flowing through a heavy-metal (HM) underlayer due to the injection of spins (whose polarization is transverse to the direction of both spin and charge current) at the FL-HM interface (assuming spin-Hall effect [10] to be the dominant underlying physical phenomenon). Such HM induced MTJ switching has been proven to be more energy efficient than STT induced switching [11, 12]. Further, it opens up the possibility of device structures that can exhibit decoupled "write" and "read" current paths, as will be explained in later sections. It is also worth noting here that at non-zero temperature, the magnetization dynamics of the MTJ is characterized by thermal noise, which can be accounted for by an additional thermal field. In the presence of thermal noise, the switching behavior of the MTJ due to the flow of a charge current through the "pinned" layer, during the "write" cycle, is stochastic in nature and the probability of switching increases with increase in the magnitude of input current [13].

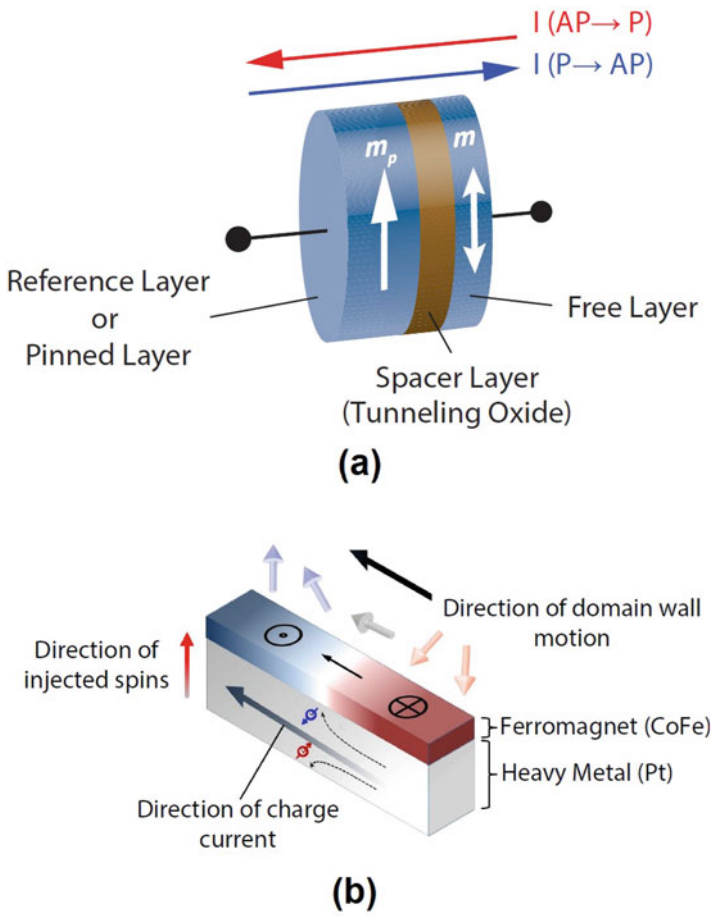

Fig. 12.3 (**a**) Magnetic tunnel junction consists of two nanomagnets separated by a spacer, (**b**) spin-orbit torque induced domain wall motion due to charge current flowing through a heavy-metal (HM) underlayer

In addition to monodomain magnets discussed above, we will also utilize multi-domain magnets having a domain wall separating oppositely polarized magnetic domains. Recent experiments on magnetic nanostrips of Pt/CoFe/MgO and Ta/CoFe/MgO have revealed high domain wall velocities due to charge current densities that are two orders of magnitude lower than that achievable by conventional spin-transfer torque (STT) [14, 15]. Additionally, domain wall motion was observed to be against the direction of electron flow (i.e., in the direction of current flow) in multilayer structures with Pt as the underlayer, thereby suggesting that current induced spin-orbit torque is the main mechanism of domain wall motion in such multilayer structures (with negligible contribution from conventional STT). In such magnetic heterostructures with high perpendicular magnetocrystalline

anisotropy (PMA), spin orbit coupling and broken inversion symmetry leads to the stabilization of homochiral domain walls through the Dzyaloshinskii-Moriya exchange interaction (DMI). Such interfacial DMI at the FM-HM interface leads to the formation of a Néel domain wall with specific chirality. The DMI strength in such structures with HM underlayers has been observed to be sufficiently strong to impose a Néel wall configuration in FMs where conventional magnetostatics would have yielded a Bloch configuration [14, 15]. As shown in Fig. 12.3b, when an in-plane charge current is injected through the HM, a transverse spin-current is generated due to deflection of opposite spin-polarizations on the top and bottom surfaces of the HM due to spin-Hall effect. The accumulated spins at the FM-HM interface leads to DMI stabilized Néel domain wall motion [14, 15]. The direction of domain wall motion is in the direction of charge current flow and the final magnetization of the ferromagnet is given by the cross-product of the direction of injected spins at the FM-HM interface and the magnetization direction of the FM at the domain wall location.

12.5 Proposals for Spintronic Neuromimetic Devices

The building blocks of the brain (neurons and synapses), as well as artificial models thereof, are fundamentally different from the switching functions and Boolean logic gates that CMOS transistors naturally realize. The significant disparity between the brain and corresponding CMOS implementations results in area and power consumptions that are orders of magnitude higher than that involved in the brain. For example, almost 20 transistors would be required to implement the functionality of a single analog spiking neuron [16]. On the other hand, digital neurons would require area and power hungry multipliers and adders to compute the weighted summation of inputs. The situation is even worse for a synapse, where storing even a single-bit weight would require a 6-T/8-T SRAM cell [17]. To realize networks comprising billions of neurons and connectivity levels exceeding 10,000 synapses per neuron, nanoelectronic devices that can more naturally and efficiently mimic synaptic and neural functionalities are imperative.

Recent discoveries in spintronics have brought forward a set of device phenomena that can provide a direct mapping of neuronal and synaptic computations, laying the foundation for a quantum leap in the efficiency of neuromorphic computing. Consider the computational units in an ANN. Inputs to the neuron get multiplied by stored synaptic weights and are subsequently summed up and passed through a thresholding function. As shown in Fig. 12.4, the synaptic functionality can be implemented using a device structure composed of a Magnetic Tunnel Junction (MTJ) where the "free" layer is a domain wall motion based magnetic strip (DWM). A DWM is a ferromagnet with oppositely polarized magnetic domains separated by a transition region termed as domain wall (DW). In the device shown in Fig. 12.4, the position of the domain wall encodes the conductance of the MTJ, which represents the synaptic weight. Therefore, the read current represents the product of the input (read voltage) and synaptic weight (conductance).

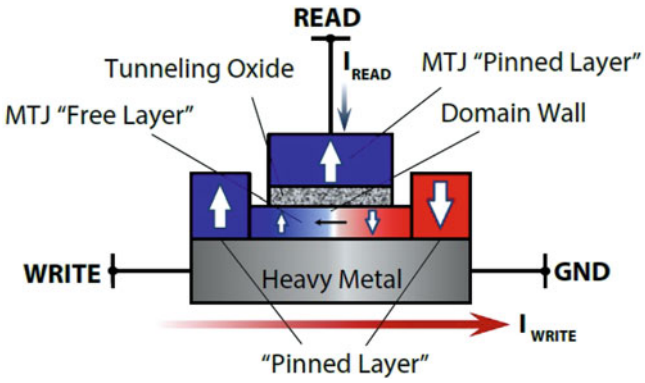

Fig. 12.4 Magnetic bilayer structure (DW-MTJ) as a spintronic synapse

As mentioned in the previous section, recent experiments on ferromagnet-heavy metal (HM) bilayers have provided a promising mechanism for efficient control of domain wall (DW) motion using current densities that can be $\sim$100$\times$ lower than conventional spin-transfer torque driven DW motion [18]. Inspired by this development, the proposed device (shown in Fig. 12.4) is a magnetic heterostructure that also includes a Heavy Metal (HM) where a current flowing through the HM can be used for deterministic control of DW motion and hence, the MTJ conductance [18]. We will refer to this device as DW-MTJ for the rest of this chapter and demonstrate its application to realizing neural and synaptic units.

Let us next consider the neuron computation in ANNs, which involves summation of synaptic inputs and performing a thresholding operation on the result. Figure 12.5 [18] shows how the DW-MTJ device discussed above, together with a reference MTJ and a single transistor, can be used to realize this computation. The DW-MTJ together with the Reference MTJ (Fig. 12.5) forms a resistive divider network. Input synaptic current is fed into the HM layer, moves the domain wall, and changes the MTJ conductance, thereby causing a variation in the output current provided by the transistor, which represents the neuron output.

A more biologically realistic neural computing model in comparison to the artificial neuron is the spiking neuron model. Such a neuron receives input spikes and generates an output spike only when its membrane potential crosses a threshold. The neuron's membrane potential increases on the arrival of an input spike and leaks back to its resting potential in the absence of a spike. Interestingly, the magnetization dynamics of an MTJ offers a direct mapping to the functionality of such spiking neurons (Fig. 12.6). Incoming synaptic current flowing from the "free" to the "pinned" layer gets spin-polarized by the "pinned" layer and causes the "free" layer magnetization to be oriented parallel to the "pinned" layer. As shown in Fig. 12.6, when input spikes are transmitted to the MTJ, the magnetization or the conductance of the MTJ starts "integrating." However, upon removal of the stimulus the magnetization starts "leaking" back. The MTJ conductance increases

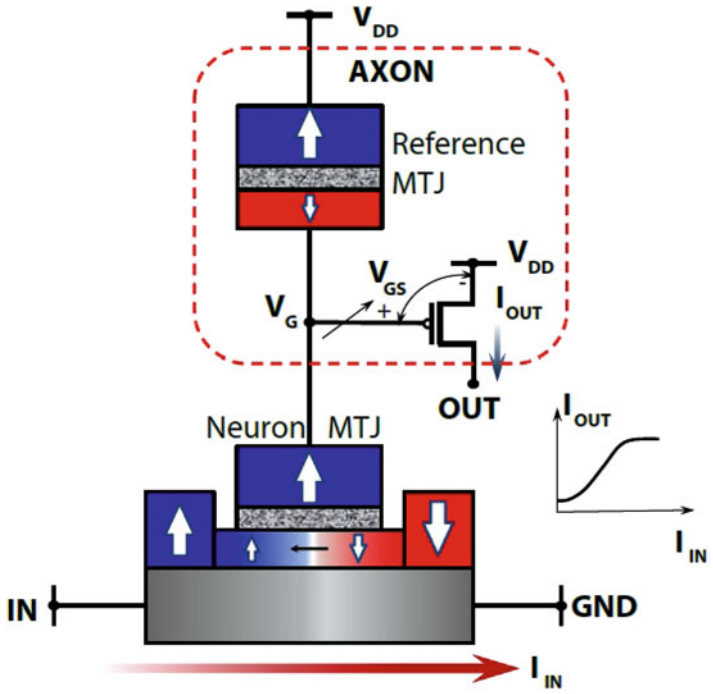

Fig. 12.5 Spintronic device (DW-MTJ) as an artificial neuron

by a specific amount for each spike and finally switches to the parallel (high-conductance) state at the end of the fifth input spike (Fig. 12.6) (analogous to the "spiking" of a biological neuron). Such a functionality can be exploited to build MTJ based spiking neurons. Further, thermal noise inherent in such devices can be exploited to perform probabilistic inference with stochastic spiking neurons [19, 20]. The MTJ switches probabilistically depending on the magnitude of the input synaptic current (Fig. 12.7). HM induced MTJ switching can significantly further improve the energy efficiency of this process.

12.6 Crossbar based "In-Memory" Computing Architecture

While the proposed spintronic devices realize the primitive functions required to implement individual neurons and synapses, realizing a multi-layer network where spin-neurons and synapses are cascaded requires hybrid circuits that involve spin devices and a few CMOS transistors. Furthermore, to support the broad range of neuromorphic applications, there is a need to design computing fabrics that can realize networks of varying sizes and topologies, and can perform both training/learning and evaluation. In doing so, it is critical to ensure that the intrinsic efficiency of spin neurons and synapses is preserved at the system level.

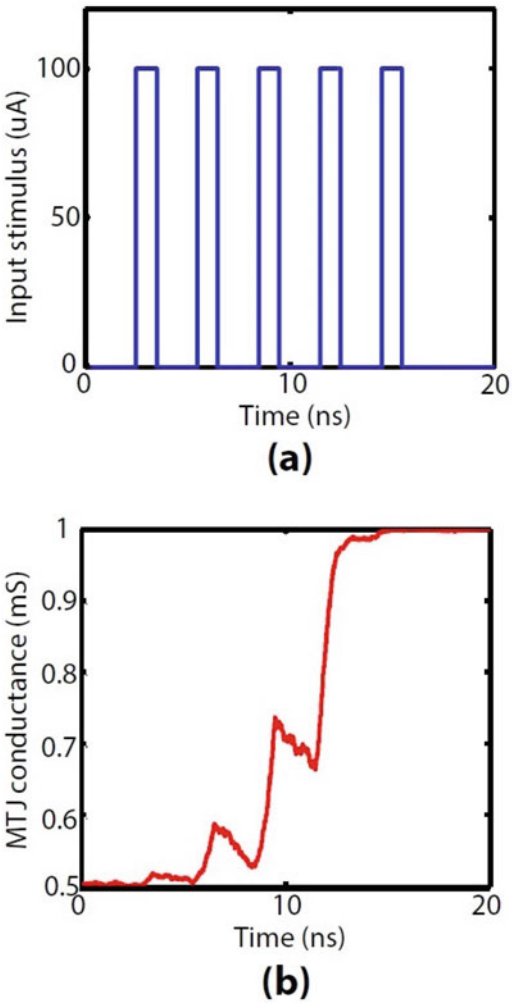

Fig. 12.6 MTJ as a leaky-integrate-fire spiking neuron. The magnetization or conductance integrates on each spike and starts leaking once the spike is removed (**a**) Input current stimulus and (**b**) MTJ conductance have been shown as a function of time

Note that the main computing kernel in a DLN is dot-product computation between the inputs and corresponding synaptic weights for each neuron followed by neuron processing. Figure 12.8 shows a possible design of such a spintronic computing kernel that consists of a crossbar array of spin-synapses driving spin-neurons. Input voltages applied across each row of the array result in the generation of a weighted synaptic current (weights are spin-synapse conductances), which is summed up and provided as input to each neuron along the column. CMOS transistors operate as axons (gate voltage is modulated by a resistive divider network consisting of a reference MTJ and the DW-MTJ). Note that the proposed processing unit can be cascaded (the drain of each output transistor drives a row of the next

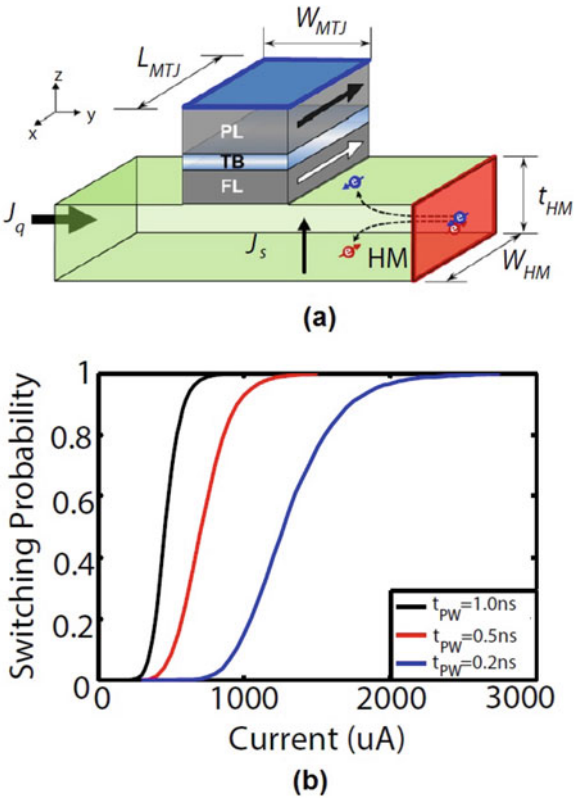

Fig. 12.7 (**a**) Probabilistic neural inference can be performed by spin-Hall effect induced MTJ switching in presence of thermal noise, (**b**) variation of MTJ switching probability with magnitude of input current for different pulse width durations

crossbar array). Hence, this unit can be used as a building block to construct scalable neuromorphic architectures and assists in implementing the dot-product computing kernel that is an essential component of such neuromimetic algorithms.

Let us analyze the spin-based ANN design shown in Fig. 12.8 and determine its advantages over a CMOS based implementation. In order to provide an intuitive insight to the energy efficiency of the proposed system, let us consider a "spin-neuron" with "free layer" dimensions of 80 nm × 20 nm. Micromagnetic simulations indicate that a current of ∼10.6 μA can displace the domain wall between the two extreme edges within 2 ns leading to a maximum energy consumption of 0.1 fJ (including energy consumption during neuron "reset" operation). This is almost two orders of magnitude lower in comparison to analog (∼700 fJ) and digital (∼832.6 fJ) CMOS neuron designs in 45 nm technology [18]. Additionally, the synaptic resistive crossbar array can be operated at ultra-low voltages of ∼100 mV due to the low current requirements of the spin-neurons. In contrast, the crossbar arrays have to be operated at a much higher voltage (∼500 mV) to drive analog CMOS neurons. This results in power (V^2/R) savings by a factor ∼25× per synapse and thereby helps in reducing the overall power consumption of the neuromorphic system.

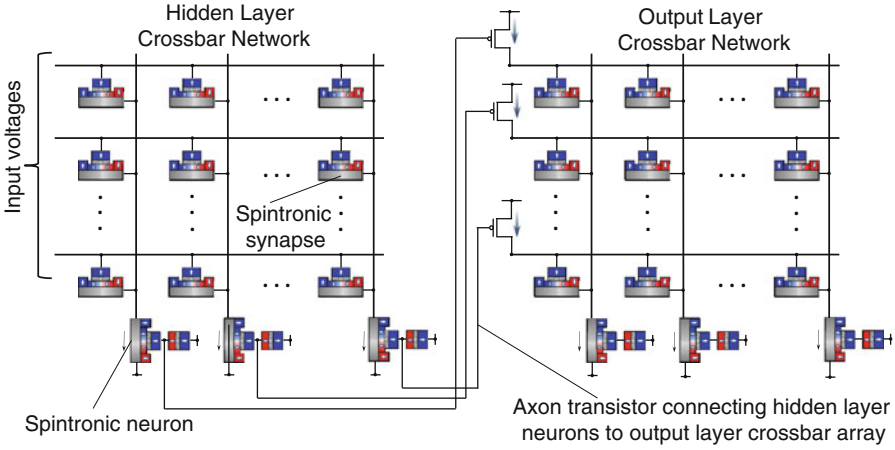

Fig. 12.8 Spintronic neuromorphic architecture (with DW-MTJ neurons and synapses) connecting different layers of the neural network

12.7 Conclusions

In this chapter, we provided a vision for "in-memory computing" architectures built on spintronic crossbars for neuromorphic computations. Crossbars alleviate the memory-bandwidth associated performance limitations in DLN acceleration. Additionally, it also removes the energy consumption associated with continuous data transfer between a separate memory and the computation core which is a dominant component of energy consumption in such data-intensive applications. Inner-product computing kernels based on spintronic crossbar arrays driving magneto-metallic spintronic neurons can pave the way for compact and energy-efficient neuromorphic architectures.

References

1. K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition (2014). arXiv preprint arXiv:1409.1556
2. T. Mikolov, M. Karafiát, L. Burget, J. Cernockỳ, S. Khudanpur, Recurrent neural network based language model.Interspeech **2**, 3 (2010)
3. Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2324 (1998)
4. A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, in *Advances in Neural Information Processing Systems* (2012), pp. 1097–1105
5. Y. Taigman, M. Yang, M. Ranzato, L. Wolf, Deepface: closing the gap to human-level performance in face verification, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2014), pp. 1701–1708

6. M. Rhu, N. Gimelshein, J. Clemons, A. Zulfiqar, S.W. Keckler, vDNN: Virtualized deep neural networks for scalable, memory-efficient neural network design (2016). arXiv preprint arXiv:1602.08124
7. Y. Chen, T. Luo, S. Liu, S. Zhang, L. He, J. Wang, L. Li, T. Chen, Z. Xu, N. Sun et al., Dadiannao: A machine-learning supercomputer, in *Proceedings of the 47th Annual IEEE/ACM International Symposium on Microarchitecture* (IEEE Computer Society, 2014), pp. 609–622
8. M. Julliere, Tunneling between ferromagnetic films. Phys. Lett. A **54**(3), 225–226 (1975)
9. J.C. Slonczewski, Conductance and exchange coupling of two ferromagnets separated by a tunneling barrier. Phys. Rev. B **39**(10), 6995 (1989)
10. J. Hirsch, Spin hall effect. Phys. Rev. Lett. **83**(9), 1834 (1999)
11. C.-F. Pai, L. Liu, Y. Li, H. Tseng, D. Ralph, R. Buhrman, Spin transfer torque devices utilizing the giant spin Hall effect of tungsten. Appl. Phys. Lett. **101**(12), 122404 (2012)
12. L. Liu, C.-F. Pai, Y. Li, H. Tseng, D. Ralph, R. Buhrman, Spin-torque switching with the giant spin Hall effect of tantalum. Science **336**(6081), 555–558 (2012)
13. A. Sengupta, S.H. Choday, Y. Kim, K. Roy, Spin orbit torque based electronic neuron. Appl. Phys. Lett. **106**(14), 143701 (2015)
14. S. Emori, U. Bauer, S.-M. Ahn, E. Martinez, G.S. Beach, Current-driven dynamics of chiral ferromagnetic domain walls. Nat. Mater. **12**(7), 611–616 (2013)
15. S. Emori, E. Martinez, K.-J. Lee, H.-W. Lee, U. Bauer, S.-M. Ahn, P. Agrawal, D.C. Bono, G.S. Beach, Spin Hall torque magnetometry of Dzyaloshinskii domain walls. Phys. Rev. B **90**(18), 184427 (2014)
16. G. Indiveri, A low-power adaptive integrate-and-fire neuron circuit, in *ISCAS (4)*. Citeseer (2003), pp. 820–823
17. P.A. Merolla, J.V. Arthur, R. Alvarez-Icaza, A.S. Cassidy, J. Sawada, F. Akopyan, B.L. Jackson, N. Imam, C. Guo, Y. Nakamura et al., A million spiking-neuron integrated circuit with a scalable communication network and interface. Science **345**(6197), 668–673 (2014)
18. A. Sengupta, Y. Shim, K. Roy, Proposal for an All-Spin Artificial Neural Network: emulating neural and synaptic functionalities through domain wall motion in ferromagnets, in *IEEE Transactions on Biomedical Circuits and Systems* (2016)
19. A. Sengupta, M. Parsa, B. Han, K. Roy, Probabilistic deep spiking neural systems enabled by magnetic tunnel junction. IEEE Trans. Electron Dev. **63**(7), 2963–2970 (2016)
20. A. Sengupta, P. Panda, P. Wijesinghe, Y. Kim, K. Roy, Magnetic tunnel junction mimics stochastic cortical spiking neurons. Sci. Rep. **6**, 30039 (2016)