coherence and GPU implementations will be slow due to the lack of coalesced memory transactions and the need to serialize warps when members of a warp take different branches. In some cases, these issues can be mitigated by partitioning the examples into groups that all take the same branch, and processing these groups of examples simultaneously. This can be an acceptable strategy for minimizing the time required to process a fixed amount of examples in an offline setting. In a real-time setting where examples must be processed continuously, partitioning the workload can result in load-balancing issues. For example, if we assign one machine to process the first step in a cascade and another machine to process the last step in a cascade, then the first will tend to be overloaded and the last will tend to be underloaded. Similar issues arise if each machine is assigned to implement different nodes of a neural decision tree.

12.1.6 Specialized Hardware Implementations of Deep Networks

Since the early days of neural networks research, hardware designers have worked on specialized hardware implementations that could speed up training and/or inference of neural network algorithms. See early and more recent reviews of specialized hardware for deep networks (Lindsey and Lindblad, 1994; Beiu *et al.*, 2003; Misra and Saha, 2010).

Different forms of specialized hardware (Graf and Jackel, 1989; Mead and Ismail, 2012; Kim *et al.*, 2009; Pham *et al.*, 2012; Chen *et al.*, 2014a,b) have been developed over the last decades, either with ASICs (application-specific integrated circuit), either with digital (based on binary representations of numbers), analog (Graf and Jackel, 1989; Mead and Ismail, 2012) (based on physical implementations of continuous values as voltages or currents) or hybrid implementations (combining digital and analog components). In recent years more flexible FPGA (field programmable gated array) implementations (where the particulars of the circuit can be written on the chip after it has been built) have been developed.

Though software implementations on general-purpose processing units (CPUs and GPUs) typically use 32 or 64 bits of precision to represent floating point numbers, it has long been known that it was possible to use less precision, at least at inference time (Holt and Baker, 1991; Holi and Hwang, 1993; Presley and Haggard, 1994; Simard and Graf, 1994; Wawrzynek *et al.*, 1996; Savich *et al.*, 2007). This has become a more pressing issue in recent years as deep learning has gained in popularity in industrial products, and as the great impact of faster hardware was demonstrated with GPUs. Another factor that motivates current research on specialized hardware for deep networks is that the rate of progress of a single CPU or GPU core has slowed down, and most recent improvements in