Scalable Energy-Efficient, Low-Latency Implementations of Trained Spiking Deep Belief Networks on SpiNNaker

Evangelos Stromatias*, Daniel Neil‡, Francesco Galluppi†, Michael Pfeiffer‡,
Shih-Chii Liu‡ and Steve Furber*
*Advanced Processor Technologies Group, School of Computer Science, University of Manchester, UK
M13 9PL, Manchester, United Kingdom
†Equipe de Vision et Calcul Naturel, Vision Institute, Université Pierre et Marie Curie
UMR S968 Inserm, UPMC, CNRS UMR 7210, CHNO des Quinze-Vingts, Paris, France
‡Institute of Neuroinformatics, University of Zurich and ETH Zurich
Winterthurerstrasse 190, CH-8057 Zurich, Switzerland

Abstract—**Deep neural networks have become the state-of-the-art approach for classification in machine learning, and Deep Belief Networks (DBNs) are one of its most successful representatives. DBNs consist of many neuron-like units, which are connected only to neurons in neighboring layers. Larger DBNs have been shown to perform better, but scaling-up poses problems for conventional CPUs, which calls for efficient implementations on parallel computing architectures, in particular reducing the communication overhead. In this context we introduce a realization of a spike-based variation of previously trained DBNs on the biologically-inspired parallel SpiNNaker platform. The DBN on SpiNNaker runs in real-time and achieves a classification performance of 95% on the MNIST handwritten digit dataset, which is only 0.06% less than that of a pure software implementation. Importantly, using a neurally-inspired architecture yields additional benefits: during network run-time on this task, the platform consumes only 0.3 W with classification latencies in the order of tens of milliseconds, making it suitable for implementing such networks on a mobile platform. The results in this paper also show how the power dissipation of the SpiNNaker platform and the classification latency of a network scales with the number of neurons and layers in the network and the overall spike activity rate.**

I. Introduction

In recent years, Deep Learning architectures such as Deep Belief Networks (DBNs) [1], [2] and Convolutional Networks [3] have surpassed all previous benchmarks in common machine learning tasks, including visual classification and speech recognition [4], and have thus been named one of the breakthrough technologies of the decade [5]. The performance of these networks can be increased by increasing the size of the networks, i.e. using networks with more layers, as described by theoretical results showing that adding more layers can only improve performance bounds [2]. Networks with large number of neurons and layers have very high computational demands, and training state-of-the-art deep networks can easily take multiple days, even on very large computer clusters [6], therefore calling for hardware accelerations, either through GPUs, or custom chips [7]. Even the execution of a trained network on standard PCs is expensive due to the large number of neurons involved, and results in high energy demands, communication overhead, and high response latencies. In particular, the long latency response is a problem for real-time applications in mobile and robotic systems, which have limited computing resources and power but require quick system responses.

One way to overcome these issues was recently demonstrated through a transformation of DBNs into spiking neural networks [8], which provide low-latency and energy efficient solutions that can be optimally implemented on event-based neuromorphic hardware platforms. The previous study developed the theory of training spiking DBNs and provided a proof-of-concept software implementation with very good performance numbers, leading to an implementation on an event-driven Field-Programmable Gate Array (FPGA) called Minitaur [9]. Here we present a more efficient implementation of this architecture on the SpiNNaker platform, a hardware platform optimized for scalable event-based simulations [10]. This platform has a biologically-inspired architecture designed to enable low-power and low-latency massively parallel large-scale simulations of heterogeneous models of spiking neurons in real-time.

We present as contributions a proof of concept spiking DBN running in real-time on a single SpiNNaker chip achieving a classification accuracy of 95% on the MNIST dataset [3], almost identical to a reference software simulator, while dissipating 0.3 W, with a mean classification latency of 20 ms. Furthermore, we characterise the latencies and power requirements of the SpiNNaker platform relevant to spiking DBNs and derive a power estimation model to predict the scalability, in terms of energy requirements, of larger DBNs running on

This work is partially supported by the European Union under grant nos. FP7-604102 (HBP), FP7-287701 (BrainScales-Extension), FP7-ICT-270324 (SeeBetter) and ERC-320689 (BIMPC), EPSRC grant EP/G015740/1 (BIMPA), SNF Grant 200021_146608, and the Samsung Advanced Institute of Technology.

larger SpiNNaker machines.

The paper is structured as follows: Section II introduces the theory behind spike-based DBNs. Section III describes the SpiNNaker architecture and the mapping of DBN parameters between software and SpiNNaker implementation. Section IV presents the results from the use of the DBN on classification of digits from the MNIST dataset [3] and evaluates the classification accuracy and latencies, as well as the power dissipation of spike-based DBNs running on SpiNNaker. Section V discusses related work of spiking RBMs and DBNs. Lastly, Section VI concludes with a discussion regarding advantages of this approach and future implementations.

II. Spike-Based Deep Belief Networks

DBNs are multi-layered neural networks, in which each layer pair is formed by a Restricted Boltzmann Machine (RBM), a recurrent network with full connectivity between two layers of visible and hidden units, but without connections between neurons of the same layer. Each neuron unit is a stochastic binary neuron, whose "on" probability is given by the weighted sum of its inputs and passed through a sigmoid nonlinearity. Training is performed layer-by-layer using the unsupervised Contrastive Divergence (CD) rule [2]. After training one layer, the outputs of the hidden units of the previous layer are used as the input to the subsequent layer. At the topmost level, a label is jointly trained with the input to give a supervised signal that guides the output of the network.

Several methods exist for training spiking versions of a neural network. In this work, we follow the formulation developed in [8]. This method is remarkably similar to the traditional method given in [2] and variations such as persistent contrastive divergence [11]. To perform training, only one change is made to the standard learning procedure: the activation function of the neuron is altered to account for the non-idealities of a spiking neuron. The so-called Siegert approximation [12] will yield the expected firing rate of a neuron given input firing rates, input weights, and standard neuron parameters. By normalizing the output firing rate of the Siegert function by the maximum firing rate of the neuron, the Siegert function can be viewed as just an abstract nonlinear activation function, real-valued between zero and one. After training, the parameters and weights can be used directly in a spiking neural network in which they produce the firing rates which match those in the training. Since these firing rates match, the implementation on biologically-realistic spiking neurons then is able to achieve the same accuracy as the rate-based network. Further details can be found in [8].

The LIF neuron model follows the membrane potential dynamics

$$\tau_m \frac{dV}{dt} = E_L - V + R_m I(t) \quad , \tag{1}$$

where τ_m is the membrane time constant set to 5 s, V is the membrane voltage, E_L is the resting potential set to 0 mV, R_m is the membrane resistance set to 1 Ω and I is the input current from the synapses (see Eq. 2). When the membrane voltage surpasses a threshold value (V_{th} = 1 mV) a spike is fired, and the membrane potential resets to V_{reset} = 0 mV. A neuron is not allowed to fire a second spike for a refractory period of T_{refrac} = 2 ms.

Equation 2 describes the synapse model used for this work:

$$I(t) = \sum_{i=0}^{n} \bar{w} \sum_{j=0}^{m_i} \delta(t - t_{ij}) \quad , \tag{2}$$

where $\bar{w}$ is the synaptic weight, n is the total number of synapses to a neuron, m_i the number of spikes received from the i^{th} synapse, and $\delta(x)$ is the Dirac delta function, which is 0 everywhere except for x = 0.

III. The SpiNNaker Platform

SpiNNaker [10] is a biologically-inspired Application Specific Integrated Circuit (ASIC) designed to enable large-scale simulations of heterogeneous models of spiking neurons in real-time. The SpiNNaker chip consists of 18 identical fully-programmable ARM9 cores. Each core has its own local 96 kB tightly-coupled memory (TCM) for data and instructions, while all the cores access a 128 MB SDRAM, through an asynchronous network on chip (NoC), where the synapses are stored.

At the centre of the SpiNNaker chip lies the router, which is responsible for communicating the spikes to the local cores or to any of the 6 bi-directional links, through an asynchronous Communications NoC. The router has been designed to handle one-to-many communication of spikes efficiently. Spikes are transmitted as multicast (MC) packets implementing the Address Event Representation (AER) scheme [13]. Each MC packet consists of a control byte that holds information about the packet type, emergency routing and time stamp information and a 32 bit routing key where the address of the neuron that fired is stored. Every SpiNNaker core has a communication controller whose purpose is to send and receive MC packets to and from the router through the Communications NoC.

By combining several SpiNNaker chips together larger machines are created. Fig. 1 shows a SpiNNaker board with 48 chips (SpiNN-5). This is the largest prototype system currently available and it will be the building block for constructing larger SpiNNaker machines. The final SpiNNaker machine will comprise approximately 1,200 of these boards, which is over a million ARM9 cores, and it aims at simulating one billion neurons with trillions of synapses in real-time. For this work the ARM9 clocks have been configured to 150 MHz, routers and system buses to 100 MHz, while the off-chip SDRAM memory clocks to 133 MHz.

The software on SpiNNaker consists of two parts, the software that runs on SpiNNaker and on the host side. On the SpiNNaker side, each core runs an application run-time kernel (SARK) with two threads that share the processor's time and are responsible for queueing and executing the tasks. The SpiNNaker Application Programming Interface (API) is built on top of SARK and allows users to develop neural and synapse models using sequential C code and by assigning callback functions that respond to particular system events [14].

Some of these events are hardware interrupts generated by one of the two hardware timers, that are available on each SpiNNaker core, and are used to solve the neural equations (Timer event). By the communications controller upon the receipt of a MC packet (Packet received event) to initiate DMA transfer in order to fetch the synapses from the SDRAM to the local memory, and by the DMA controller (DMA done event) to inform the core that the synapses have been copied to the local TCM.

On the host side, pyNN [15], a simulator-independent neural specification language, is used to describe neural topologies and their parameters using abstractions such as populations and projections. Partition And Configuration MANagement (PACMAN) [16] a tool that is based on a pyNN script, maps a spiking neural network to a SpiNNaker machine based on the available resources.

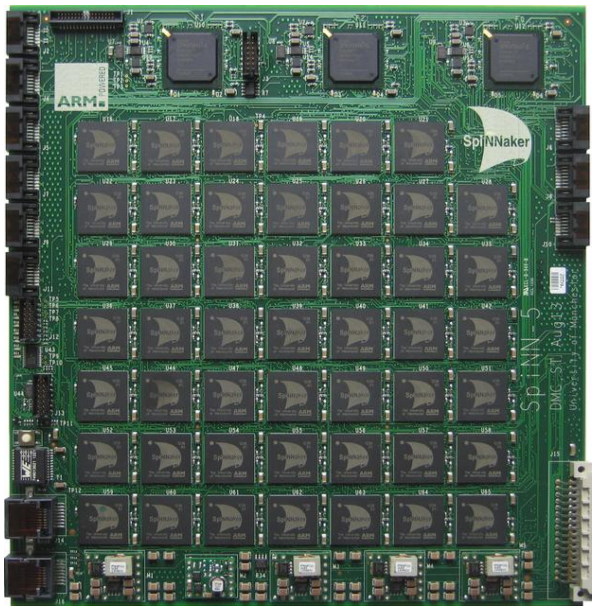

Fig. 1. A SpiNNaker board with 48 chips (SpiNN-5).

To implement spiking DBNs on SpiNNaker, a collection of functions were developed in Python that read a MATLAB file of an off-line trained DBN [8] and automatically generate a pyNN description of that network ready to run on SpiNNaker. The same network can also be tested on the Brian [17] spiking neural network simulator as a method to verify the classification performance of spiking DBNs on SpiNNaker.

By taking advantage of the re-programmability of the SpiNNaker cores we increased the precision of our weight model using a fixed-point accuracy of 22 bits, 6 for the integer part and 16 fractional part (Q6.16), while the resolution for synaptic delays was decreased from 4 to 1 bit as delays do not play an important role in the current implementation.

IV. Results

A. Spike-Based Deep Belief Networks on SpiNNaker

We chose the MNIST dataset [3] as a classification benchmark. The MNIST dataset consists of 28×28 gray-scale pixel images of handwritten digits and is divided into a training set, which comprises 60,000 digits, and a test set, which consists of 10,000 digits. The DBN topology is a 784-500-500-10 network trained off-line as described in [8].

Static images are transformed into spike trains by converting each pixel of an MNIST image into a Poisson spike-train with a rate proportional to its intensity, while all firing rates are scaled such that the total firing rate of the population is constant [8]. Fig. 2 shows the topology and firing rates of each layer of the spiking DBN for an MNIST digit running on SpiNNaker. We chose the classification accuracy (CA) as our metric of performance, which is the percentage of correctly classified digits over the whole MNIST test set. We compared the performance of the SpiNNaker implementation against the software implementation in MATLAB and Brian [17]. Where possible we also compare these results against those on Minitaur [9], an FPGA event-based implementation of the same topology network.

Both the MATLAB implementation and the Brian simulator employ double floating-point arithmetic and achieved a CA of 96.06% and 95.07% respectively. In SpiNNaker the weights are represented using 16 bits for the fractional part (Q6.16), while Minitaur uses 11 bits (Q5.11). SpiNNaker achieved a CA of 95.01%, while Minitaur achieved 92% (see Table I). The results indicate that there is only a 1% loss in performance when switching to spiking neuron models, which is in accordance with a previous study [8]. Furthermore, our SpiNNaker implementation with reduced weight precision achieves almost equivalent performance as a spiking software model. The difference in performance between the two hardware platforms is likely due to the weight precision; the quantized look-up table Minitaur uses for the membrane decay; and the event driven update of Minitaur.

TABLE I
Classification accuracy (CA) of the same DBN running on different platforms.

Simulator	CA (%)	Description
Matlab	96.06	Rate-based (Siegert)
Brian	95.07	Clock-driven
SpiNNaker	95.01	Hybrid
Minitaur	92.00	Event-driven

B. Classification Latencies

In order to investigate the real-time performance of spike-based DBNs running on SpiNNaker, two sets of experiments are conducted. The first experiment measures the classification latency of the spiking DBN as implemented on SpiNNaker. The second experiment investigates how the mean classification latency and accuracy are affected by the total number of input spikes. The increase in classification latency as the network scales up is also estimated in this experiment.

In order to measure the classification latency of a spike-based DBN running on SpiNNaker in the first experiment, a Tektronix TDS 3034B oscilloscope is used to measure the time from the first input spike to the first output spike by recording

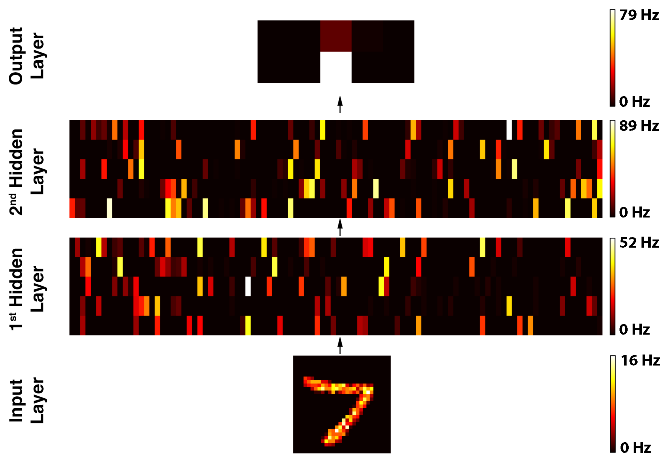

Fig. 2. The topology and firing rates of the 784-500-500-10 spiking DBN under investigation for a single input MNIST digit. The bottom plot shows firing rates of the 28×28 neurons in the input population. The next two rows of 5×100 show the firing rates of the neurons in the first and second hidden layer (500 neurons each), and finally the top plot shows the firing rates of the 10 neurons in the output population, one for each digit from 0 to 9. The arrows indicate all-to-all connections, which means that a neuron in one layer connects to all neurons in the next layer.

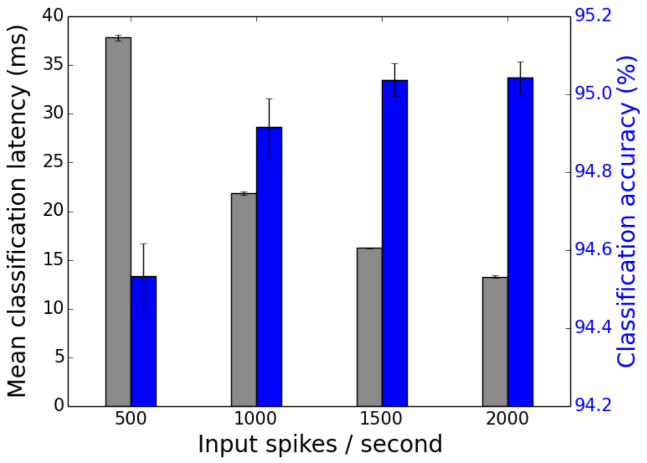

Fig. 4. Mean classification latency and classification accuracy as a function of the input spikes per second. As firing rates increase, the classification latency drops and accuracy increases. Results are averaged over 4 trials, error bars show standard deviations.

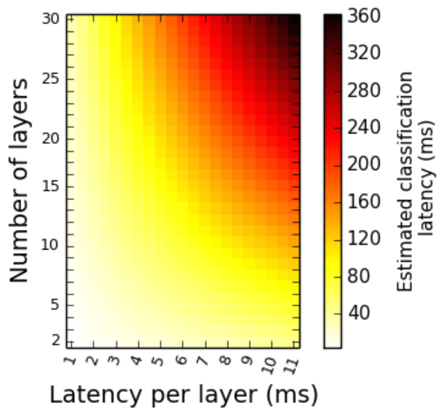

Fig. 5. Estimated classification latencies of larger spiking DBNs as a function of the total number of hidden layers and the latency per hidden layer.

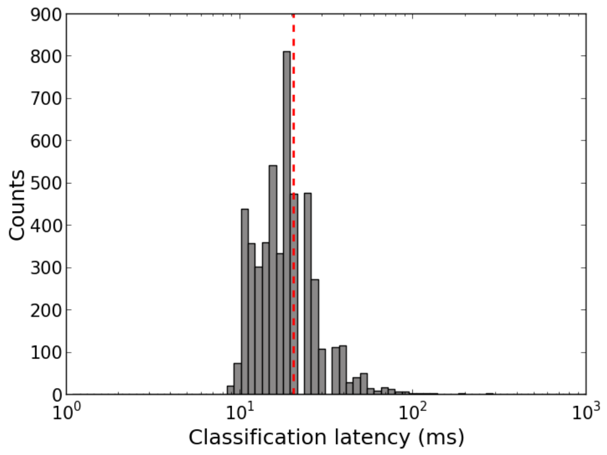

Fig. 3. Histogram of the classification latencies for the first 5,000 MNIST digits of the test set using 100 bins. The mean classification latency is shown as a red dashed line. Results are from the SpiNNaker implementation.

the signals from the general-purpose input/output (GPIO) pins of the SpiNNaker board. The results as seen in Fig. 3 show a mean classification latency of 20 ms. A latency in the order of ms is expected since the timer events used to solve the neuron equations for the experiments are set to 1 ms, which is the default SpiNNaker configuration.

Two additional experiments are carried out using the Brian software simulator to investigate how the number of input spikes affects the mean classification latency and to estimate the classification latencies of larger DBNs. For the first experiment, the static images of the MNIST test digits are converted to spike-trains using rates between 500 to 2,000 spikes per second. Results in Fig. 4 show that increasing the number of input spikes reduces the mean classification latency, and has a positive effect on the classification accuracy of the spiking DBN.

The latency of the hidden layers also varies with the number of input spikes. For the MNIST test set encoded with 2,000 spikes per second the mean spike latency between the hidden layers is 2.3 ms, while when 500 spikes per second are used the mean latency per hidden layer rises to 6.22 ms. A classification latency estimation model is developed in order to predict the mean classification latencies of larger spiking DBNs for a number of fixed values of latencies per layer. Results are summarised in Fig. 5.

C. Power Requirements

The power requirements of the SpiNNaker platform as the size of the network scales up is explored here. Two sets of experiments are conducted: The first experiment aims at characterising the energy requirements of a SpiNNaker chip and deriving a power estimation model. The second experiment investigates the power requirements of a spiking DBN running on a single SpiNNaker chip and utilises the power estimation model to explore the scalability of spiking DBNs on larger SpiNNaker machines in terms of energy requirements.

1) Power estimation model of SpiNNaker: The same methodology in measuring the power is used as in [18], but with measurements on a board with a single SpiNNaker chip (see Fig. 6). This setup allows for a more accurate energy measurement, and these measurements are then used to build a power estimation model for the SpiNNaker platform based on the number of neurons and the number of synaptic events per second. A 0.05 Ω shunt resistor is placed in series with the 1.2V voltage regulator that supplies voltage to the SpiNNaker chip, and an 1.0 Ω resistor is placed in series with the 1.8V voltage regulator that supplies the SDRAM and the inputs/outputs of the chip (Fig. 6). A BK Precision 2831E voltage meter is used to measure the voltage drop across the resistors (sampled once per second) which is proportional to the current flowing into the chip.

The benchmark neural network used in the power characterisation task comprises 16 populations of LIF neurons, recurrently connected in an all-to-all fashion, using delta-current synapses (Eq. 2), Fig. 7. The activity of the network is controlled through a single parameter, the current injected directly into the neurons, while all synaptic weights are set to zero so they would not alter the dynamics of the network. Whenever a neuron's membrane potential reaches its threshold value, it fires a spike (MC packet) and the router routes it back to the originating core. Despite the fact that all synapses are zero-valued the same process takes place upon the receipt of a MC packet, that is initiating a look-up process, starting a DMA request to fetch synapses from the SDRAM to the local memory and updating the status of each synapse. Table II summarises the neural parameters used during these experiments.

TABLE II
PARAMETERS OF THE LIF NEURONS. VALUES INSIDE THE BRACKETS INDICATE THE RANGE OF A UNIFORM DISTRIBUTION FROM WHICH PARAMETER VALUES ARE DRAWN.

Parameters	Values	Units
τ_m	64.0	mV
V_{init}	$[-65.0, -125.0]$	mV
V_{reset}	$[-90.0, -125.0]$	mV
V_{thresh}	$[-50.0, -60.0]$	mV
τ_{refract}	3.0	ms

The power estimation model of a SpiNNaker chip can be described as:

$$P_{\text{tot}} = P_I + P_B + (P_N \times n) + (P_S \times s) \qquad (3)$$

where P_I is the power dissipated by a SpiNNaker chip after the booting process with no applications loaded, P_B is the baseline power which consists of the power dissipated by SARK and

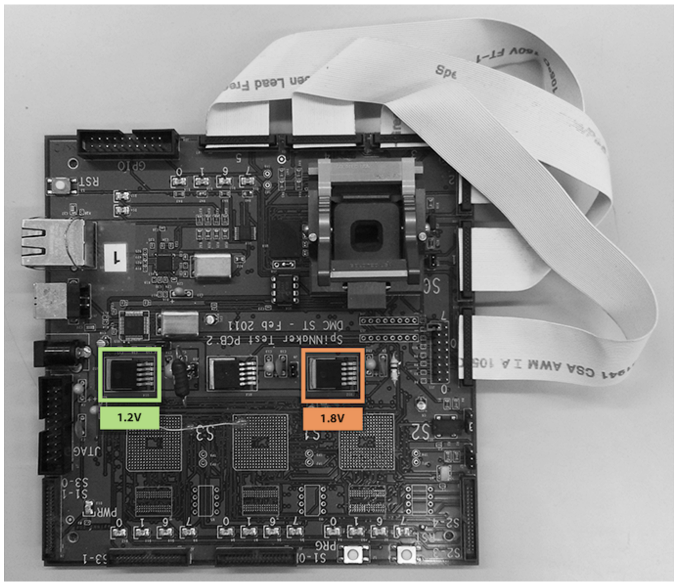

Fig. 6. A board with a single SpiNNaker chip attached to a zero insertion force (ZIF) socket. The 1.2V voltage regulator (green color) supplies power to the SpiNNaker chip and the 1.8V voltage regulator supplies power to the inputs/outputs of the chip and the SDRAM.

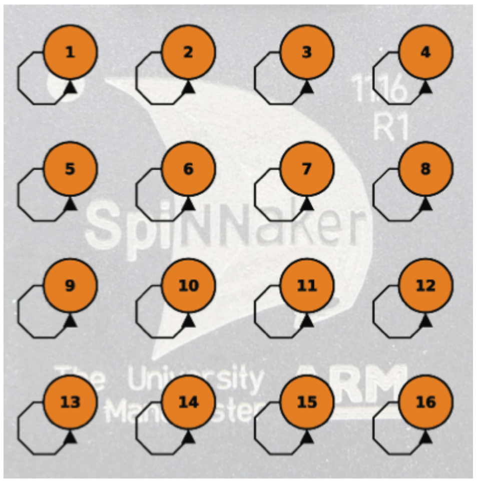

Fig. 7. The benchmark neural network used to derive a power estimation model. This neural architecture comprises 16 populations of LIF neurons self-connected in an all-to-all fashion. The activity of the network is controlled through the current injected to the neurons.

TABLE III
EXPERIMENTAL RESULTS OF THE SPINNAKER POWER ESTIMATION MODEL.

Parameters	Values	Units	Description
P_I	0.25	W	Idle power
P_B	4.0	mW	Baseline power
P_N	23.4	µJ	Energy to simulate an LIF per ms
P_S	11.3	nJ	Energy per synaptic event

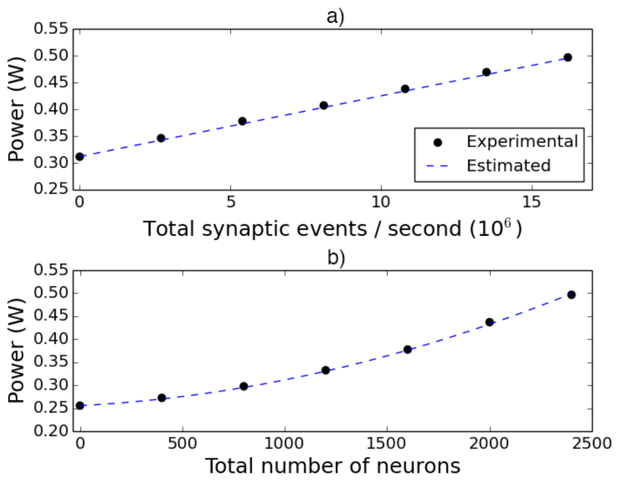

Fig. 8. Power dissipation of a SpiNNaker chip. a) Real and estimated power dissipation when the number of neurons is fixed to 2,400 and total synaptic events per second are varied. b) Real and estimated power dissipation when the total synaptic events are fixed and the number of neurons is varied from 0 to 2,400.

the SpiNNaker API without running any neurons on it, P_N is the power required to simulate an LIF neuron with a 1 ms time-step, n are the total number of neurons, P_S is the energy consumed per synaptic event (activation of neural connections) and s are the total synaptic events.

In order to calculate the power required to simulate an LIF neuron (P_N), the benchmark network is simulated on SpiNNaker. Each population consists of 150 neurons and the injected current is set to 2.45 nA in order to generate a mean firing rate of 46 Hz per population, while the spike transmission was disabled. Table II summarises the neuron parameters used during the experiments. The difference between the power when running the benchmark network simulation, and the sum of P_I and P_B is P_N. For calculating the energy consumed per synaptic event, the spike transmission is re-enabled and P_S is the difference between this simulation and the aforementioned one. Results are summarised in Table III.

Two additional experiments are conducted in order to validate the accuracy of the power estimation model using the benchmark neural topology. For the first experiment 2,400 LIF neurons are used and the total number of synaptic events per second are varied from 0 to 16,560,000. Fig. 8a shows the power dissipated by a single SpiNNaker chip (dots) during the experiments and the estimated ones (dashed line) by utilising Eq. 3. For the second experiment the total synaptic events per second are kept constant and the number of neurons are varied from 0 to 2,400. Fig. 8b shows the power dissipated by a SpiNNaker chip (black dots) and the estimated ones (dashed line).

When the SpiNNaker cores are clocked at 150 Mhz, a SpiNNaker chip can simulate 2,400 LIF neurons (150 per core) and handle up to 16,560,000 synaptic events per second (1,035,000 per core), while fulfilling the real-time restriction (ms updates) and dissipating 0.5 W.

2) Power dissipation of spike-based DBNs on SpiNNaker: In order to investigate the power requirements of a spike-based DBN running on a single SpiNNaker chip, the same methodology is employed as in the previous subsection, Fig. 6. The simulation ran for 10 seconds, while the number of input spikes generated for the same MNIST digit varied from 0 to 2000 spikes per second. Results show that both the power dissipation and number of output spikes increase with the number of input spikes per digit (Fig. 9). When 2000 spikes per second are used per digit, a SpiNNaker chip dissipates 0.3 W, and that accounts for simulating 1794 LIF neurons with an activity of 1,569,000 synaptic events per second. For the identical spiking DBN implemented on Minitaur which is clocked at 75 Mhz, a power dissipation of 1.5 W is reported when 1000 spikes per image were used [9].

A final experiment is carried out in order to investigate the power requirements of larger spiking DBNs running on a SpiNNaker board with 48 chips, Fig. 1. The power estimation equation, Eq. 3, is used to estimate the dependence of the power dissipation on the total number of hidden layers and neurons per hidden layer. Three different firing rates are assumed for the neurons in the hidden layer, 10 Hz, 15 Hz and 20 Hz. The results are summarised in Fig. 10. The power estimation model is used to estimate the power under two different criteria: The minimum number of chips required to simulate the total number of neurons based on the number of hidden layers and neurons per layer of the spiking DBN (2,400 neurons per SpiNNaker chip), and the minimum amount of chips required to support the total number of synaptic events per second (16,560,000 per SpiNNaker chip). The white area in Fig. 10 signifies the parameter regimes where real-time simulation is not feasible because the total synaptic events per second require more than 48 SpiNNaker chips. Results show that for different topologies of spiking DBNs, the limiting factor is the number of synaptic events as the number of hidden layers goes up, while the estimated power dissipation of spiking DBNs utilising a full 48 chip board is less than 22 W.

V. Comparison With Related Work

Investigations of spike-based DBNs are still rare with most of the reported studies carried out on a two-layered RBM. Exceptions so far are the software DBN model in [8], and the hardware implementation in [9]. The MNIST database was frequently used to determine the classification accuracy of the network. The software DBN of size 728-1000-500-300-50 by Eliasmith et al. [19] achieved 94% classification accuracy. The network used rate-based neurons except for the final output layer which was implemented with spiking neurons due to limitations on the available computing resources.

Neftci et al. [20] recently proposed an event-based variation of an online CD rule to train spiking RBMs. The trained two-layer software spike-based RBM with 824 visible neurons and 500 hidden neurons achieved a classification accuracy of 91.9%. Petrovici et al. [21] implemented spike-based RBMs

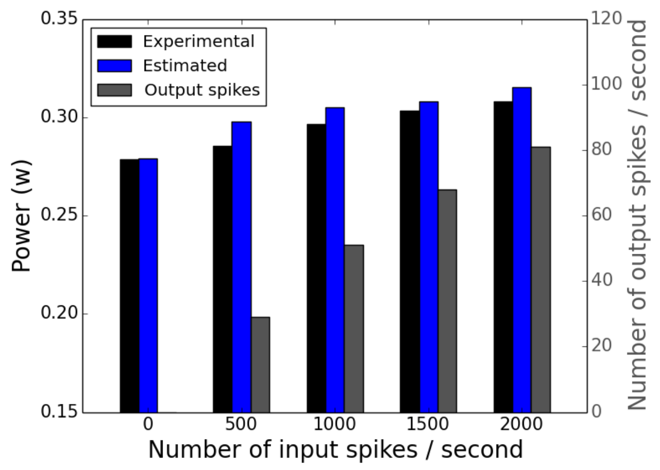

Fig. 9. Real and estimated power dissipation of a spike-based DBN running on a single SpiNNaker chip as a function of the number of input spikes generated for the same MNIST digit. The right axis shows the number of output spikes as a function of the number of input spikes. The left bars (0 input spikes) shows power dissipation when the network is idle.

consisting of LIF neurons, following the theoretical framework of neural sampling [22]. However, no results for the MNIST dataset are available for this approach.

Arthur et al. [23] trained a two-layer RBM consisting of 484 visible and 256 hidden units, and 10 linear classifiers in the output layer to classify the MNIST digits. The RBM was then mapped to spiking neurons by utilising a global inhibitory rhythm over fixed time windows [24]. A hardware implementation of their digital neurosynaptic core, which contains 256 LIF neurons simulated at discrete time-steps of 1 ms, led to a classification accuracy of 89% at an energy consumption of 45 pJ. The current TrueNorth chip [25] consists of 4,096 such cores and has a maximum capacity of 1 million neurons, which can be simulated in real time.

VI. Conclusions and Future Work

In this paper we present the most efficient implementation of spike-based DBNs to date, running on the biologically-inspired massively-parallel fixed-point SpiNNaker platform. Its architecture is optimized for simulations with massive parallelism, asynchronous updates, and event-based chip-to-chip communication. It is an excellent fit for simulating the stereotypical neural updates and relatively sparse connections of deep networks in real-time and with minimal power consumption. Combining spiking neural networks and this hardware platform is thus an ideal fit for mobile or robotics applications [26], which require fast responses while interacting with the environment, and have only a limited power budget compared to currently popular GPU- or cloud-based solutions.

Our simulations verify that the advantages in energy efficiency come at only small, tolerable inaccuracies. The loss in performance due to the limitation of fixed-point arithmetic

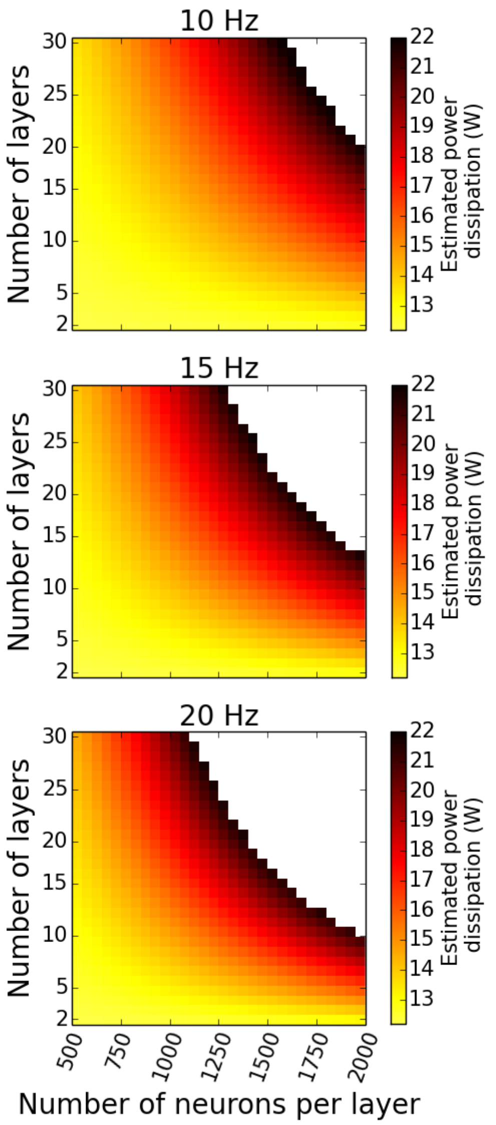

Fig. 10. Estimating the power requirements of larger spiking DBNs running on a 48 chip SpiNNaker board, as a function of the number of hidden layers and neurons per layer, for three different firing rates for the neurons in the hidden layers. The white area denotes regimes where real-time simulation is impossible due to excessive synaptic events per second.

on SpiNNaker when compared to a double precision floating-point MATLAB implementation, is only around 1% on the MNIST dataset of handwritten digits, which is in agreement with previous studies [8]. The difference is almost negligible (0.06%) compared to a similar implementation in the Brian spiking neuron simulator.

The classification latencies of the implemented spiking DBNs on SpiNNaker are in the order of tens of milliseconds, which is fast enough for interactive real-time classification. Additionally, we have demonstrated that as the number of input spikes increase, the classification accuracy improves while latency decreases.

The power consumption for the spiking DBN under test running on a single SpiNNaker chip is less than 0.3 W for a digit encoded with a rate of 2000 spikes per second, while it is estimated that larger DBNs running on a larger prototype SpiNNaker board will dissipate less than 22 W.

Another very promising feature of our architecture is its scalability: it is well known that adding more layers to DBNs can improve performance [2]. With the SpiNNaker architecture it becomes possible to create very large DBNs by adding additional layers, running on different cores or chips, without significantly increasing the latency of the system, and at reasonable power dissipation. Future work will thus explore implementations on larger hardware platforms, such as the currently largest prototype comprised of 48 SpiNNaker chips.

While the SpiNNaker hardware might not achieve the energy performance of dedicated neuromorphic hardware, the programmability of its architecture makes it an excellent exploration platform for event-based computing. Future work will aim at interfacing neuromorphic sensors [27], [28], [29] as the input layer of spiking DBNs, or experiment with possible plasticity rules [30] which can be implemented in real-time for on-line unsupervised learning of RBMs [20].

Acknowledgement

The authors would like to acknowledge the 2013 Telluride Workshop on Neuromorphic Engineering where the initial work was carried out and then continued at the 2014 Capo Caccia Cognitive Neuromorphic Engineering Workshop.

References

[1] G. Hinton and R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[2] G. Hinton, S. Osindero, and Y. W. Teh, "A fast learning algorithm for deep belief nets." *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[3] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[4] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85–117, 2015.

[5] R. Hof. (2013) 10 breakthrough technologies of 2013. [Online]. Available: http://www.technologyreview.com/featuredstory/513696/deep-learning/

[6] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, M. Mao, A. Senior, P. Tucker, K. Yang, Q. V. Le *et al.*, "Large scale distributed deep networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 1223–1231.

[7] C. Farabet, B. Martini, B. Corda, P. Akselrod, E. Culurciello, and Y. LeCun, "Neuflow: A runtime reconfigurable dataflow processor for vision," in *CVPR Workshops*. IEEE, 2011, pp. 109–116.

[8] P. O'Connor, D. Neil, S.-C. Liu, T. Delbruck, and M. Pfeiffer, "Real-time classification and sensor fusion with a spiking deep belief network," *Frontiers in Neuroscience*, vol. 7, no. 178, 2013.

[9] D. Neil and S.-C. Liu, "Minitaur, an event-driven fpga-based spiking network accelerator," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. PP, no. 99, pp. 1–1, 2014.

[10] S. Furber, F. Galluppi, S. Temple, and L. Plana, "The SpiNNaker Project," *Proceedings of the IEEE*, vol. 102, no. 5, pp. 652–665, 2014.

[11] T. Tieleman and G. Hinton, "Using fast weights to improve persistent contrastive divergence," in *Proc. of ICML*. ACM, 2009, pp. 1033–1040.

[12] F. Jug, J. Lengler, C. Krautz, and A. Steger, "Spiking networks and their rate-based equivalents: does it make sense to use Siegert neurons?" in *Swiss Soc. for Neuroscience*, 2012.

[13] M. Mahowald, *An Analog VLSI System for Stereoscopic Vision*. Norwell, MA, USA: Kluwer Academic Publishers, 1994.

[14] T. Sharp, L. Plana, F. Galluppi, and S. Furber, "Event-driven simulation of arbitrary spiking neural networks on spinnaker," in *Neural Information Processing*, ser. Lecture Notes in Computer Science, B.-L. Lu, L. Zhang, and J. Kwok, Eds. Springer Berlin Heidelberg, 2011, vol. 7064, pp. 424–430.

[15] A. P. Davidson, D. Bruderle, J. M. Eppler, J. Kremkow, E. Muller, D. Pecevski, L. Perrinet, and P. Yger, "PyNN: a common interface for neuronal network simulators," *Frontiers in Neuroinformatics*, vol. 2, pp. 1–10, 2009.

[16] F. Galluppi, S. Davies, A. Rast, T. Sharp, L. A. Plana, and S. Furber, "A hierarchical configuration system for a massively parallel neural hardware platform," in *Proceedings of the 9th Conference on Computing Frontiers*, ser. CF '12. New York, NY, USA: ACM, 2012, pp. 183–192.

[17] D. F. Goodman and R. Brette, "Brian: a simulator for spiking neural networks in python," *Frontiers in Neuroinformatics*, vol. 2, no. 5, 2008.

[18] E. Stromatias, F. Galluppi, C. Patterson, and S. Furber, "Power analysis of large-scale, real-time neural networks on spinnaker," in *IJCNN*, 2013, pp. 1–8.

[19] C. Eliasmith, T. C. Stewart, X. Choo, T. Bekolay, T. DeWolf, Y. Tang, and D. Rasmussen, "A large-scale model of the functioning brain," *Science*, vol. 338, no. 6111, pp. 1202–1205, 2012.

[20] E. Neftci, S. Das, B. Pedroni, K. Kreutz-Delgado, and G. Cauwenberghs, "Event-driven contrastive divergence for spiking neuromorphic systems," *Frontiers in Neuroscience*, vol. 7, no. 272, 2014.

[21] M. A. Petrovici, J. Bill, I. Bytschok, J. Schemmel, and K. Meier, "Stochastic inference with deterministic spiking neurons," *arXiv preprint*, vol. 1311.3211, 2013.

[22] L. Buesing, J. Bill, B. Nessler, and W. Maass, "Neural dynamics as sampling: a model for stochastic computation in recurrent networks of spiking neurons," *PLoS Computational Biology*, vol. 7, no. 11, p. e1002211, 2011.

[23] J. Arthur, P. Merolla, F. Akopyan, R. Alvarez, A. Cassidy, S. Chandra, S. Esser, N. Imam, W. Risk, D. Rubin, R. Manohar, and D. Modha, "Building block of a programmable neuromorphic substrate: A digital neurosynaptic core," in *Neural Networks (IJCNN), The 2012 International Joint Conference on*, June 2012, pp. 1–8.

[24] P. Merolla, T. Ursell, and J. V. Arthur, "The thermodynamic temperature of a rhythmic spiking network," *CoRR*, vol. abs/1009.5473, 2010. [Online]. Available: http://arxiv.org/abs/1009.5473

[25] 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, B. Brezzo, I. Vo, S. K. Esser, R. Appuswamy, B. Taba, A. Amir, M. D. Flickner, W. P. Risk, R. Manohar, and D. S. Modha, "A million spiking-neuron integrated circuit with a scalable communication network and interface," *Science*, vol. 345, no. 6197, pp. 668–673, 2014.

[26] F. Galluppi, C. Denk, M. Meiner, T. Stewart, L. Plana, C. Eliasmith, S. Furber, and J. Conradt, "Event-based neural computing on an autonomous mobile platform," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2014, pp. 2862–2867.

[27] S.-C. Liu, A. van Schaik, B. Minch, and T. Delbruck, "Event-based 64-channel binaural silicon cochlea with q enhancement mechanisms," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2010, pp. 2027–2030.

[28] P. Lichtsteiner, C. Posch, and T. Delbruck, "A 128$\times$128 120 db 15 μs latency asynchronous temporal contrast vision sensor," *Solid-State Circuits, IEEE Journal of*, vol. 43, no. 2, pp. 566–576, Feb 2008.

[29] J. Leñero Bardallo, T. Serrano-Gotarredona, and B. Linares-Barranco, "A 3.6 μs latency asynchronous frame-free event-driven dynamic-vision-sensor," *IEEE Journal of Solid-State Circuits*, vol. 46, no. 6, pp. 1443–1455, June 2011.

[30] F. Galluppi, X. Lagorce, E. Stromatias, M. Pfeiffer, L. A. Plana, S. B. Furber, and R. B. Benosman, "A framework for plasticity implementation on the spinnaker neural architecture," *Frontiers in Neuroscience*, vol. 8, no. 429, 2014.