Spiking Binary Associative Memory for Implementing on Neuromorphic Hardware

Shamini Koravuna
Research Institute for Cognition and Robotics (CoR-Lab)
Bielefeld University, Bielefeld, Germany

Prof. Dr. -Ing. Ulrich Rückert
Research Institute for Cognition and Robotics (CoR-Lab)
Bielefeld University, Bielefeld, Germany

I. MOTIVATION

The aim of the project is the design space exploration of embedded hardware platforms for the simulation of Spiking Neural Networks (SNNs), which on the one hand, allows resource-efficient execution, and, on the other hand, facilitates online adaptation and learning. The exploration of the neuromorphic accelerators focuses on reconfigurable hardware platforms (Field Programmable Gate Arrays (FPGAs)).

The basis is an integrated design process based on a backend-independent representation to describe the SNN architecture, learning algorithm, neuron and synapse model that automatically maps SNNs to parameterizable hardware architectures. The developed hardware platforms are coupled with event-based sensors (e.g., DVS cameras) as part of a prototype structure and tested and evaluated using practical application scenarios from two domain experts from vision-based quality control and event detection.

II. PROBLEM STATEMENT

Many application areas, for example, computer vision tasks in production, require resource-efficient execution. Artificial Intelligence solutions that adapt to online and changing conditions in such methods are promising but often involve large models and costly computations. Real-time processing is necessary for many applications that cannot handle latency in making important decisions. Also, the applications could be using private data that could be sensitive if sent to the cloud. By performing all computation with local data in the edge and thus following a privacy by design methodology, this project contributes to the field of trustworthy AI.

III. METHOD

The project uses current embedded hardware architecture for testing SNN specific online learning methods directly on hardware. As an example, this system will be employed for ultra-high-speed computer vision and event detection tasks. Implementing online learning methods into resource-efficient hardware allows embedding such tasks directly on sensor hardware, reducing high-bandwidth communications, allows faster processing, and ensures data privacy. Firstly, concerning hardware architecture, this will cover reconfigurable hardware platforms (FPGA's). Secondly, various neuron and synapse models – and configurations of these models – will be evaluated, for example, variations of spike-timing-dependent plasticity (STDP) [1]. The hardware architecture will finally be coupled with event-based sensors (e.g., DVS cameras) to evaluate the solutions based on practical application scenarios.

The research carries out a backend-independent model exploration (e.g., described by ONNX [2], SpineML [3] or NineML [4]) of SNN network architecture, learning algorithm and neuron/synapse model, which determines the first parameters for the subsequent exploration of the hardware platform. This first evaluation is intended to determine the complexity of the various neuron/synapse models to decide which model properties (e.g., refractory times, adaptive thresholds, different variants of STDP) are required and optional for the hardware implementation. STDP forms a component of the adaptive platform that can adapt to changing data (data or concept drift). Another building block consists of the optimization of the network architecture by reducing the network during inactivity (analogous to the "nocturnal reduction" when sleeping), i.e., removing the most insignificant synapses and creating new connections (structural plasticity [5]). The abstract model is mapped to the target architecture by a model compiler. The network structure is optimized independently of the architecture in preparation, i.e., clusters of neurons with high connectivity are physically placed close to one another to find the Pareto optimum trade-off between communication costs and calculation costs. The synaptic delay of the spiking neurons can be used to efficiently implement the delivery of spikes to more distant neurons in hardware. Both the clustering of neurons with high connectivity and the use of varying delays can be integrated into the learning process (see, e.g., Spike Timing Dependent Delay Adaptation [6] or SpikeProp, a form of spike-based backpropagation [7]). Alternatively, delays are specified by the hardware platform and are integrated into the learning algorithm [9]. The complete design flow for implementing the neuromorphic hardware is depicted in Figure 1.

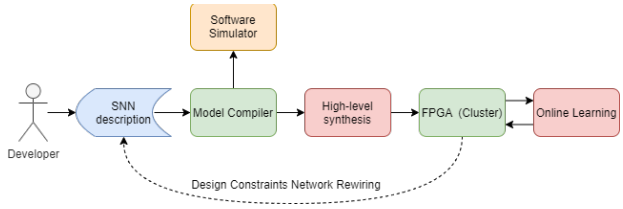

Figure 1: Design Flow for Implementing Neuromorphic Hardware

IV. BINARY ASSOCIATIVE MEMORY

As the research focuses on exploring hardware architectures, memory plays a vital in hardware designing. For the first assessment of the influence of neuron/synapse model complexity on application performance, we make use of a simplistic spiking binary associative memory model. Associative memory is a mapping memory that maps input to output patterns. The binary associative memory was developed by G. Palm [8] in 1980, and the spiking variant is inspired from this. Binary associative memory consists of a single layer of neurons, in which every neuron represents one output element. The weights of the connection matrix have to be trained according to the input and output patterns. To get a

reasonable capacity, we must have sparse patterns in the input and the output.

The spiking network architecture is realized using a Leaky-Integrate and Fire (LIF) neuron model. In the input vector, ones/set bits are encoded as a single spike. For decoding the output, spikes in a given time window are interpreted as a set bit in the respective output pattern. When the pre-trained storage matrix M_{xy} evaluates to 1, a synaptic connection is established between input signal 'x' and neuron 'y'. All the inputs connected with the weights are summed up, and are compared with the threshold; if the values exceed the threshold, the neuron fires. If the neuron does not fire it relaxes back to the resting potential. In contrast to the classical associative memory, the spiking variant has several neuron parameters that need to be optimized for a good performance. In this case, and to better understand the underlying principles, this is realized by multi-dimensional parameter sweeps.

V. RESULTS

For implementing the SNN model architecture, the Brian2 software simulator is used. The input and output patterns are generated in a way to be sparse to attain reasonable capacity. We investigated how the storage behaves with varying samples and other essential model parameters like threshold and weights. The essential model parameters are determined by performing multiple experiments. Default values for all plots can be found in Table 1. Figure 2 shows the behavior of the storage with a varying number of samples. This demonstrates, that the performance of the spiking variant relative to the classical realization and the size of the storage capacity is proportional to the number of samples.

Table 1: Default Parameters

Synaptic Weight (nA)	1
Threshold (mV)	4
Input Neurons	300
Output Neurons	600
Ones in Input	4
Ones in Output	5

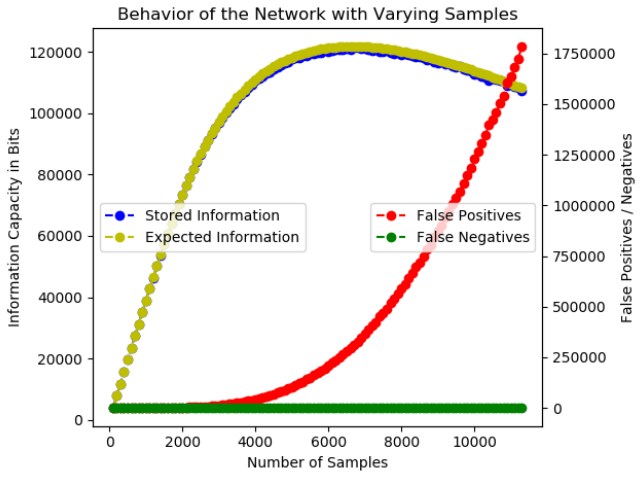

Figure 2: Number of Samples vs Information

The storage performance is estimated at varying thresholds and weights with the inputs and outputs specified in Table 1. As an example, Figures 3 depicts the model's performance at varying weight values. On one hand, small weights lead to information loss (false negatives) in the output. On the other hand, increased weights will result in a large number of false positives. For optimal performance, the weight has to be chosen such that both values are at a minimum.

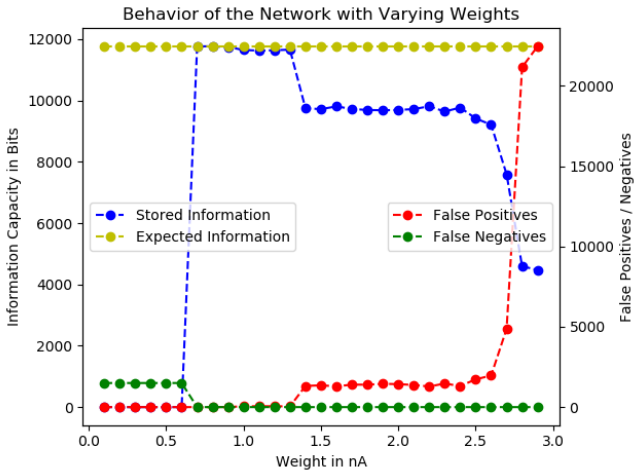

Figure 3: Weight vs Information

VI. CONCLUSION

To summarize, a simple binary associative memory is realized using SNNs. The essential parameters are determined. For future work, the model will be implemented on neuromorphic hardware (FPGAs) for drawing associations in real-world edge AI applications. And the results will be compared with the existing neuromorphic hardware architectures.

REFERENCES

[1] Taherkhani, Belatreche, Li, Cosma, Maguire, and McGinnity: "A review of learning in bio-logically plausible spiking neural networks", Neural Networks, 122, pp. 253–272. 2020.

[2] https://github.com/onnx/onnx and https://onnx.ai/

[3] Richmond, Cope, Gurney, and Allerton: "From Model Specification to Simulation of Bio-logically Constr. Networks of Spiking Neurons", Neuroinformatics, 12, pp. 307–323, 2013.

[4] Raikov, Cannon, Clewley, Cornelis, Davison, De Schutter, Szatmary, et al.: "NineML: the network interchange for neuroscience modeling language", BMC Neuroscience, 12, 2011.

[5] Knoblauch, and Sommer: "Structural Plasticity, Effectual Connectivity, and Memory in Cor-tex", Frontiers in Neuroanatomy, 10, 2016.

[6] Wang, Hamilton, Tapson, and van Schaik: "A mixed-signal implementation of a polychro-nous spiking neural network with delay adaptation", Front. in Neuroscience, 8, 2014.

[7] Schrauwen, and Van Campenhout: "Extending SpikeProp", IEEE International Joint Con-ference on Neural Networks, 2004.

[8] Palm, On associative memory. Biol. Cybern. 36, 19–31,1980. doi: 10.1007/BF00337019

[9] Bellec, Scherr, Subramoney, Hajek, Salaj, Legenstein, and Maass: "A solution to the learn-ing dilemma for recurrent networks of spiking neurons", Cold Spring Harbor Lab., 2019.