Live Demonstration: Face Recognition on an Ultra-low Power Event-driven Convolutional Neural Network ASIC

Qian Liu[1], Ole Richter[1], Carsten Nielsen[1,2], Sadique Sheik[1], Giacomo Indiveri[1,2], and Ning Qiao[1,2]

[1]aiCTX AG, Thurgauerstrasse 40, CH-8050 Zurich, Switzerland
[2]Univeristy of Zurich, Zurich, Switzerland

Abstract

We demonstrate an event-driven Deep Learning (DL) hardware software ecosystem. The user-friendly software tools port models from Keras (popular machine learning libraries), automaticaly convert DL models to Spiking equivalents, i.e. Spiking Convolutional Neural Networks (SCNNs) and run spiking simulations of the converted models on the hardware emulator for testing and prototyping. More importantly, the software ports the converted models onto a novel, ultra-low power, real-time, event-driven ASIC SCNN Chip: DynapCNN. An interactive demonstration of a real-time face recognition system built using the above pipeline is shown as an example.

1. Introduction

There has been a tremendous growth in our understanding and use of SCNNs in the recent past [3, 4]. This has enabled us to make a significant leap in our computational abilities with Spiking Neural Networks (SNNs). SNNs also enable us to harness the power efficiency of neuromorphic engineering to build ultra-low power neural network systems. Integrating these two aspects, we demonstrate a novel real-time neuromorphic hardware software eco-system for simulating and emulating SCNNs.

We demonstrate various components of this eco-system, including the user friendly software framework - SINABS for developing and simulating SCNNs. As an application demonstration we will show how a simple 6 layer Convolutional Neural Network (CNN) trained to recognize faces of several people can be converted to its spiking equivalent SCNN. We will then show how this model processes input from an onsite Dynamic Vision Sensor (DVS) [1] and predict the identity of the person in front of the camera. We will also port the same model onto our DynapCNN and show the output and power consumption of the chip in real-time.

2. SINABS: Open-source Python library

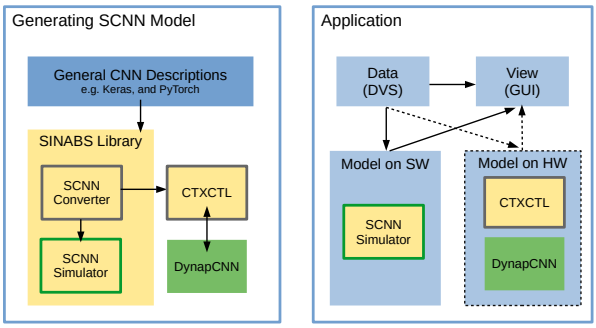

Figure 1. The architecture of the HW-SW ecosystem. Left: the process of generating an SCNN model onto a HW/SW simulator; Right: the components to run any SCNN-based applications.

SINABS library provides a smooth transition from Deep Neural Networks (DNNs) to SCNNs implementation. It allows simulation of multilayer-SCNNs. The SCNNs simulations allow emulation and validation of models in inference mode. The library also enables simulations at arbitrary fixed precision. The library is based on PyTorch and all the layers are fully compatible with all PyTorch modules. So all PyTorch functionality such as simulating on GPUs is retained. In addition, one of the core functionality of this library is to enable porting of models defined and developed in Keras into equivalent SCNNs.

Finally and most crucially, CTXCTL is a software framework aimed at facilitating the control and execution of experiments using neuromorphic hardware communicating through Address-Event Representation (AER). CTXCTL also provides visualization tools for efficient debugging and network design. CTXCTL driver module enables porting the developed models from Python onto an ultra-low power and latency DynapCNN chip. Figure 1 shows a block diagram of the various modules of the hardware sotware ecosystem and how they interact with each other.