Real-Time SNNs Model Analyzing and Visualizing Experimentation using RAVSim

Sanaullah*
University of Applied Science,
Bielefeld, Germany

Shamini Koravuna
CoR-Lab, University of Bielefeld,
Bielefeld, Germany

Ulrich Rückert
CoR-Lab, University of Bielefeld,
Bielefeld, Germany

Thorsten Jungeblut
University of Applied Science,
Bielefeld, Germany

1 INTRODUCTION

The human brain is a fascinating mystery. It controls our body which is a very complicated non-linear dynamic system, and the brain can control it at an extremely fast speed. The growth of the experimental findings and evidence that spike timing is essential to explain neural computations has motivated the use of Spiking neural networks (SNNs). However, a significant amount of dedicated simulators have been developed to analyze and visualize the SNNs behavior. Nevertheless, there are many challenges and computational issues related to SNNs. In some cases, it requires the use of accurate biological representations of the neurons. Although some of the existing state-of-the-art SNN simulators Brian2 [1] and NEST [2] have been primarily built for exploring brain functionalities and neuronal dynamics, they are not user-friendly. Adding new functionalities to some simulators requires specifying them in a low-level programming language for example C++ and integrating them with the simulator code [2]. Also, some of the simulators need domain-specific languages, for instance, NESTML for NEST simulator , NMODL for NEURON [3]. With all these constraints, there is a requirement of an interactive simulator with low or no-code as the existing simulators require a lot of time and huge lines of code for designing the neural network architectures and for analyzing and visualizing their behavior. Additionally, a fast, real time visualization, and user-friendly simulator not only speeds up existing simulations, but also accelerates the process of designing, prototyping, parameter tuning and so on. Also, research into other algorithms, including training advances the field as a whole. With this motivation behind and the existing challenges in understanding and leveraging the promising features of SNNs, we are proposing a novel real-time simulator "RAVSim", a state-of-the-art SNN simulator, implemented using LabVIEW (Laboratory Virtual Instrument Engineering Workbench) [4] and also, RAVSim v1.0 was successfully accepted and published on LabVIEW's official website [5] after being reviewed by their developer team. It is an interactive simulation environment tool that allows the user to interact with the model, observe its behavior, and make direct changes in the parameters of input models at run-time. Our proposed approach allows the SNN architecture to be defined completely in software, which negates the need to re-synthesize the hardware implementation when any parameter in the SNN architecture is changed.

2 RAVS*im* TOOL

Real-time Analyzing and Visualization Simulator or RAVSim is an interactive virtual experiment environment for the simulation and analysis of the SNN models. RAVSim tool uses Leaky integration and fire (LIF) model with continuous noise input [6], spike detection by using input current [7], and generating a winner takes all network (WTA) [8], which establishes communication between the neurons. The run-time simulation environment is important in that the input-output value is critical for extracting the correct logic behavior of an SNN model. Thus, it is necessary to visualize the value of this parameter before applying it directly to the hardware. One should be aware of which parameters are essential, how the parameters interact and are dependent on each other, the optimal values of these parameters for achieving accurate and efficient SNNs.

The real-time environment is also very important in the following sense: Let us assume that a user has concluded an analysis with some random parametric value settings (such as threshold value, reset potential, membrane capacitance, and so on). If the user conducts another analysis with different parametric value settings and the computation time in the current analysis exceeds abnormally compared to the previous analysis, the user can deduce that he/she might have selected some invalid parametric value settings for the current analysis. Below we provide some results to support the above hypothesis. But, we must first recall the definition of the LIF model: In order to perform correct analysis, it is therefore required to initialize all of the model parameters to a stable value. If the values are not initialized or entered wrong, the real-time simulator visualization feature helps the user, and that is the reason we set some

default parametric values in the start of the simulation and also using menu bar user can see the default values at any time during the simulation. The default parameter VI helps the user to specify a correct value during which the SNN model output is expected to become stable. The overall RAVSim demonstration using default parametric values shown in the [9], Supporting Information section User Manual.

3 DISCUSSION AND RESULTS

RAVSim allows the users to observe the SNN parameter reactions graphically and interact with the model in real-time. This process is analogous to setting up parameters for experimentation and testing of a model in any other programming language like Python. But, for doing so we require huge lines of code to wholly understand these parameters and the model needs to be fine-tuned with various values of the parameter just to observe their behavior and for acquiring the appropriate values. The current simulators to designing SNNs model is time-consuming in both prospectives studying and understanding, as many of times model may need to run with different parametric values, out of which only a few would function.

RAVSimulator is a particular balance between the conflicting demands of flexibility, user-friendly interface, and faster performance and compare the results of these available simulators. The RAVSim is also very helpful in the following sense: let's assume that a user has implemented 10,000 neurons based network with some random parametric value settings on the hardware and results of accuracy or whatever users are looking for output, comes after several hours of analysis and in case the user didn't satisfy the output then run the analysis again after changing the values of the parameters and so on. Additionally, this is also a time–consuming and costly procedure, costly in the sense that might be the target board (i.e. FPGA) overheated and stopped working, and most important to perform a correct experiment user needs to verify each parametric value with different combination every time, which is also a very lengthy and complex procedure. However, RAVSim real–time environment offers the user to visualize the experiment before starting a practical implementation to verify the stability of each parameter.

We have tested RAVSim v1.0 using up to 1000 neurons, as you can be seen in Supporting information section Experimental Results, in the most complex model 'i != j's case the maximum amount of time consumed by the simulator is an average of 12 seconds to analyze and visualize the WTA network. However, brain2 simulator in the same case consumed 328 seconds. The details of each experimental result can be accessed at [9], Supporting Information section Experimental Results.

4 CONCLUSION

To summarize, a simple yet effective real-time simulator, "RAVSim" has been developed, which is very interactive, faster, and easy to use for understanding the mechanism of SNNs. This tool is ideal for early-stage researchers and students. It helps them to interact with the simulator in real-time and understand the working of SNNs for designing optimal neural networks. It saves a lot of time for the users as it is an interactive tool. It does not require any programming skills and helps them learn things quickly. For future work, we are continuously working on improving RAVSim and will implement other SNN neuron and synapse models, and various learning techniques which avoid huge lines of code for simulating the models. We believe that this work (RAVSim Simulator), Real-time analysis, and visualization of the SNNs model will be an interesting area for further research.

AVAILABILITY

All of the experiments have been performed using real-time simulations on RAVSim v1.0. The RAVSim (v1.0) is an open-source simulator and available publicly at [5] and video demonstration of RAVSim can be accessed at [$https://www.youtube.com/watch?v = Ozv0MXXj89Yt = 28s$]

ACKNOWLEDGEMENTS

The authors would like to thank Christopher Ostrau from the Center for Cognitive Interaction Technology (CITEC) at Bielefeld University, for his contribution to the development of earlier versions of RAVSim. This research is supported by DataNinja Graduate School, Germany.

REFERENCES

[1] Marcel Stimberg, Romain Brette, and Dan FM Goodman. Brian 2, an intuitive and efficient neural simulator. *Elife*, 8:e47314, 2019.
[2] Jochen M Eppler, Moritz Helias, Eilif Muller, Markus Diesmann, and Marc-Oliver Gewaltig. Pynest: a convenient interface to the nest simulator. *Frontiers in neuroinformatics*, 2:12, 2009.
[3] ML Hines. ExpandingNEURON's RepertoireofMechanisms withNMODL.
[4] LabVIEW: laboratory virtual instrument engineering workbench. https://www.ni.com/de-de/shop/labview.html. Accessed: January 2022.
[5] https://www.ni.com/de-de/support/downloads/tools-network/download.real-time-analysis-and-visualization-simulator--ravsim-.html#443936.
[6] Grégory Dumont, Jacques Henry, and Carmen Oana Tarniceriu. Noisy threshold in neuronal models: connections with the noisy leaky integrate-and-fire model. *Journal of mathematical biology*, 73(6):1413–1436, 2016.
[7] Nicolas Fourcaud-Trocmé, David Hansel, Carl Van Vreeswijk, and Nicolas Brunel. How spike generation mechanisms determine the neuronal response to fluctuating inputs. *Journal of neuroscience*, 23(37): 11628–11640, 2003.
[8] Yanqing Chen. Mechanisms of winner-take-all and group selection in neuronal spiking networks. *Frontiers in computational neuroscience*.
[9] https://fh-bielefeld.sciebo.de/s/SzeBaSHfVtlHs3u.