Implementation of Leaky Integrate-and-Fire Neuron Model for Handwritten Character Recognition with a Deployment Demonstration on Jetson Nano

Akshata U. Bhomkar[1], Prof. Deeksha[2], Nisarga S. Bhat[3], Pooja M. Waingankar[4], Sudha V. Gatteppanavar[5]
[1,3,4,5]*Department of Electronics and Communication Engineering, Srinivas Institute of Technology, Mangaluru – 574 143, Karnataka, India*
[2]*Assistant Professor, Department of Electronics and Communication Engineering, Srinivas Institute of Technology, Mangaluru – 574 143, Karnataka, India*
doi.org/10.64643/IJIRTV12I12-206812-459

***Abstract*—In this paper, the design, simulation, and implementation of the LIF model are performed for recognizing handwritten characters. At first, the spike response of the neuron to different amounts of input currents was analyzed in MATLAB. Then, the accuracy of the results was verified in order to confirm the correct behavior of the neuron under different circumstances. In the next step, the same model was implemented using Python code to perform classification based on EMNIST data set, converting the pixel values to spikes in a period of time. Next, the model was run on the NVIDIA Jetson Nano Developer Kit, resulting in a working system with acceptable performance in an actual environment. As a result, it could be understood that inference is possible even in real-time mode without a significant amount of energy consumption.**

***Index Terms*—Edge AI, EMNIST dataset, Jetson Nano, Leaky Integrate-and-Fire neuron, neuromorphic computing, spiking neural networks, temporal coding.**

I. INTRODUCTION

The neuron is the unit of information processing in living organisms. Neurons exchange information by generating spikes, i.e., electrical impulses that are produced when the potential of the neuron exceeds a predefined value. Contrary to standard models, such an exchange process is highly sparse and timing-dependent, which distinguishes it from regular artificial neural networks.

The standard artificial neural network model is based on continuous variables and synchronization of the computations. Despite high applicability and good results achieved, the usage of artificial neural networks requires significant computational resources and power. As a consequence, such a network cannot be easily implemented into embedded systems due to limitations in computational capacity. These issues have led to the growing interest in Spiking Neural Networks (SNN), primarily due to its event-driven approach.

Among the existing neuron models, the Leaky Integrate- and-Fire (LIF) model is one of the simplest and most frequently applied. The LIF model accounts for such phenomena as integration of input currents, leaking over time, and spiking upon reaching the threshold level. After generating a spike, the neuron is reset, allowing the cycle to be repeated again. Despite its simplicity, the model can represent rate and temporal coding in many scenarios. The following tasks are addressed in this study:

- Investigating the performance of the LIF neuron model for various input currents with MATLAB
- Implementing the model for recognizing handwritten characters based on the EMNIST database.
- Evaluating the entire framework on Jetson Nano to verify real-time operation

II. BACKGROUND AND RELATED WORK

The Leaky Integrate-and-Fire (LIF) neuron model was presented as a simple alternative to complicated biological neuron models, such as Hodgkin–Huxley. It was shown previously [1] that implementations of LIF using MATLAB can mimic realistic spikes and yet remain computationally efficient. Another research by

Chaturvedi et al. [2] compared LIF model with the more advanced Izhikevich model and concluded that even though it is less complex, LIF may perform relatively well for character recognition provided correct parameter tuning.

In several studies, such as [3], some spike-based character recognition approaches were presented as opposed to typical training methods. Afterward, adaptive thresholding techniques proposed by Bawane [4] led to some improvement. Lu [5] suggested a method to link LIF with deep neural network architectures, thus making the training process more efficient and easier.

From the viewpoint of hardware implementation, Banik and Basu [15] showed how spiking neural networks could be used on Jetson Nano for recognizing digits. The study reveals that spike-based networks may be performed on embedded hardware devices without requiring excessive resources. In this paper, attempts are made to unify the three stages

III. THE LIF NEURON MODEL

The membrane voltage for a LIF neuron can be obtained using a first-order differential equation:

$$\tau_m \frac{dV(t)}{dt} = -V(t) + RI(t) \qquad --(1)$$

This means that the membrane voltage varies with time based on the influence of the current as well as the leakage effect.

$V(t) \geq V(th) \rightarrow$ emit spike -- (2)

When the voltage becomes equal to a particular threshold level, a spike occurs, and the voltage is reduced.

$$V(t+\Delta t) = V(t) + (\Delta t/\tau)\,[-V(t) + R \cdot I(t)] \quad --(3)$$

The behavior of the neuron is mostly determined by some key parameters:

The membrane time constant, τm, which defines how quickly the potential will drop.

Threshold potential, Vth, that determines the triggering point of the neuron's firing.

Refractory period, tref, which limits the frequency of spike generation.

The tuning of these parameters is rather straightforward, which greatly facilitates modelling and simulation.

IV. METHODOLOGY

A. MATLAB Spike Simulation

The following parameter values were used for the simulation: R = 50 MΩ, C = 1 nF, Vth = 10 mV, Vreset = 0.2Vth, and tref = 5 ms. The current input to the neuron ranged between 1 and 50 nA.

Euler's method was used to advance the membrane potential in time at intervals of 1 ms for 50 ms. A spike was counted every time the membrane potential reached the threshold voltage. Then, the potential was set to its reset value, and the neuron became inactive until the end of the refractory period.

The waveforms were visually inspected. The desired response was usually observed in the simulations, with slight variations due to different inputs.

B. Python Character Recognition

EMNIST Letters Dataset was adopted for the classification task. First, all the images were normalized to a value range between 0 and 1. Then, these numbers were scaled and considered input currents to the neuronal model.

As a rule, brighter pixels translated into stronger currents that generated more spikes per neuron. The model was trained for several time steps with collecting statistics on spikes by classes. Finally, the decision of what letter to predict was made by the class with the maximum number of spikes.

The Gradio application was built to facilitate experiments. It enables users to load or draw letters and instantly see the results.

C. Jetson Nano Deployment

The Python program was run on the Jetson Nano system (Ubuntu 20.04, Python 3.8). For the calculations part, NumPy library was utilized, whereas TensorFlow was applied for the dataset operations.

The presence of GPU enabled some parallel calculations to be performed, thus increasing performance as compared to CPU only implementation. Again, the same interface was maintained as before, enabling easier comparison of the results obtained with the earlier ones.

V. RESULTS AND DISCUSSION

A. LIF Neuron Behavior (MATLAB)

From the simulation results, it can be seen that the firing rate is proportional to the value of the current

being supplied.

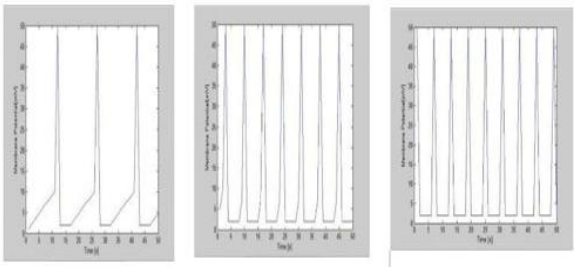

Fig.1. Membrane potential responses of the LIF neuron for synaptic input currents of (a) 1 NA, (b) 5NA, (c) 50 NA,

When the current being applied is low, there will be a decrease in the spike rate since it takes time for the membrane potential to increase.

As the value of the current is increased, the gap between the spikes will become smaller.

In instances where the value of the current is high, there is an increase in the firing rate despite its being constrained by the refractory period.

B. Image Preprocessing and Encoding

Grayscale EMNIST images of characters were read and normalized to values in the range of [0, 1]. Fig. 2 shows two such characters: A and C, along with their processed counterparts.

The normalization procedure leads to a consistent, high-contrast image, which corresponds directly to the encoding of currents for neurons: an intensity of zero in the image will correspond to an input current of zero (or silence), and an intensity of one will correspond to the highest possible current, resulting in the fastest possible spike frequency.

Fig.2. Comparison of Sample EMNIST Dataset Images and their Pre-processed Representations

C. Spike Pattern Analysis

Each neuron will produce a unique spike train.

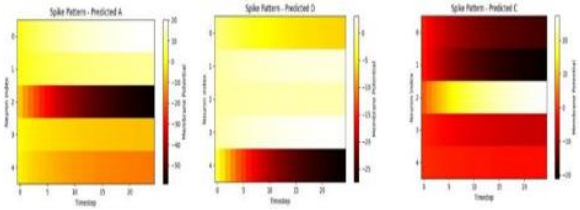

Fig.3. Spike pattern showing temporal spike activity and neuron firing distribution.

The spike trains for each example are not the same, but there is definitely a difference between the two classes. It is this difference that helps the model classify the characters, even if not perfectly.

D. Gradio Interface

Interface provides real-time testing as the input image, the processed output image, and predicted label are displayed simultaneously.

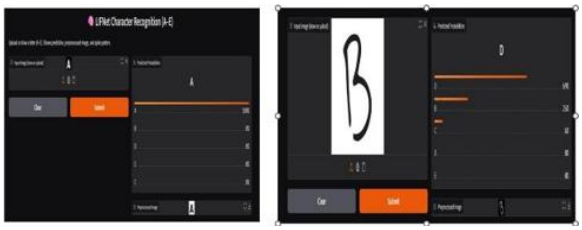

Fig.4. Gradio-based user interface showing input image, preprocessing output and predicted character using the LIF-based SNN

Offline testing revealed that predictions for the top five most frequent EMNIST letters match the ground truth in most of the trials. Most predictions have been accurate. Nevertheless, certain mistakes have occurred. Some mistakes occur for those characters that look alike, like C and G or B and D.

E. Jetson Nano Deployment

Inference could be carried out in less than a second per picture, making the system capable of being used in real-time applications. The power requirement was found to be in the normal range of about 5 to 10 W.

There were some errors in classification. This could have been caused by similarities in the characters or insufficient variety in the data set. Regardless, the system's performance was satisfactory at the prototype stage.

Fig.5. Output on the NVIDIA Jetson Nano

VI. CONCLUSION

In this project, the development of an entire system utilizing the LIF neuron was done to identify handwritten characters. The use of MATLAB simulation was helpful in analysing the behavior of the neuron. The development process utilizing Python helped understand how spike-based encoding could be used for classifying characters.

Deployment of this model on Jetson Nano implies that such models are feasible to implement on embedded devices without needing extensive resources. However, further improvements are possible. For instance, applying learning algorithms such as STDP, experimenting with deeper networks or enhancing the encoding method would be some ideas for future study.

ACKNOWLEDGMENT

The authors would like to express their sincere gratitude to Prof. Deeksha, Dr. Soorya Krishna K., and Dr. Shrinivasa Mayya D. for their valuable guidance and support throughout this study. Appreciation is also offered for the assistance rendered by the A Shama Rao Foundation through laboratory facilities.

REFERENCES

[1] S. Chouhan, "An analytical study of leaky integrate-and-fire neuron model using MATLAB simulation," *International Journal of Engineering Research & Technology (IJERT)*, vol. 2, no. 4, Apr. 2013.

[2] S. A. Chaturvedi, N. R. Sondhiya, R. N. Titre, A. A. Khurshid, and S. S. Dorle, "Comparison of LIF and Izhikevich spiking neural models for recognition of uppercase and lowercase English characters," *CiiT International Journal of Digital Image Processing*, vol. 6, no. 6, Jun.–Jul. 2014.

[3] Gupta and L. N. Long, "Character recognition using spiking neural networks," in *Proc. International Joint Conference on Neural Networks (IJCNN)*, 2007, pp. 53–58.

[4] P. Bawane, "Object and character recognition using spiking neural networks," M.Tech. thesis, Dept. Comput. Sci., 2018.

[5] S. Lu, "Linear leaky-integrate-and-fire neuron model based SNNs and deep NN mapping," *arXiv preprint*, 2022.

[6] M. Belyaev, "A spiking neural network based on the model of VO_2," *Applied Sciences*, 2019.

[7] Tavanaei and A. Maida, "Bio-inspired multi-layer spiking neural network extracts discriminative features from speech signals," in *Proc. International Conference on Artificial Neural Networks (ICANN)*, 2017.

[8] N. Russo, W. Yuzhong, T. Madsen, and K. Nikolic, "Pattern recognition spiking neural network for classification of Chinese characters," *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

[9] S. Y. A. Yarga and S. U. N. Wood, "Accelerating SNN training with stochastic parallelizable spiking neurons," *arXiv preprint*, 2023.

[10] J. Seekings, P. Chandarana, M. Ardakani, M.-R. Mohammadi, and R. Zand, "Towards efficient deployment of hybrid SNNs on neuromorphic and edge AI hardware," 2024.

[11] NVIDIA, "Get started with Jetson Nano developer kit," *NVIDIA Developer Guide*. [Online]. Available: https://developer.nvidia.com/embedded/learn/get-started-jetson-nano-devkit

[12] Rosebrock, "How to configure your NVIDIA Jetson Nano for computer vision and deep learning," *PyImageSearch*, 2020.

[13] T. Gilbert, "Jetson-Nano-Code-Collection," GitHub repository, 2024. [Online]. Available: https://github.com/TannerGilbert/Jetson-Nano-Code-Collection

[14] M. Izhikevich, "Simple model of spiking neurons," *IEEE Transactions on Neural Networks*, vol. 14, no. 6, pp. 1569–1572, Nov. 2003.

[15] Banik and A. Basu, "Spiking neural network deployment on Jetson Nano," in *Proc. IEEE CONECCT*, 2021, pp. 1–5.