Spiking Neural Network Simulation and Classification

Your Name
Department of Electrical and Computer Engineering
Drexel University
Philadelphia, PA, USA
your.email@drexel.edu

***Abstract*—This paper presents a comprehensive study of a Spiking Neural Network (SNN) simulation and classification project. The project involves simulating an SNN using a simple model with multiple layers of neurons, each having a membrane potential, voltage threshold, and decay rate. The simulation processes input spikes through the network layers and updates the neurons' states based on the input and their weights. Additionally, a small SNN is implemented to classify data with two classes, where the data is a single float value per sample. The data is converted to spikes for the classification task. This paper provides an in-depth explanation of the project's intricacies, including the main simulation code, the classification model, and the data used.**

***Keywords*—Spiking Neural Network, SNN, Simulation, Classification, Neural Network, IEEE**

I. Introduction

Spiking Neural Networks (SNNs) are a type of artificial neural network that more closely mimic the behavior of biological neurons. Unlike traditional neural networks, SNNs use spikes, or discrete events, to transmit information between neurons. This project aims to simulate an SNN and implement a small SNN for data classification.

II. Main Simulation Code

The main simulation code is implemented in `main.c`. The code initializes the network, processes input spikes through the network layers, and updates the neurons' states based on the input and their weights. The high-level approach involves the following steps:

1) **Network Initialization**: The network is initialized with a specified number of neurons per layer. Each neuron is assigned initial values for membrane potential, voltage threshold, and decay rate.
2) **Input Initialization**: Input spikes for the first layer are initialized with random values.
3) **Layer Processing**: Each layer of the network is processed for a specified number of time steps (`TAU`). During each time step, the neurons' states are updated based on the input spikes and their weights. The membrane potential of each neuron is decayed, and neurons fire if their membrane potential exceeds the voltage threshold.
4) **Output**: The output of the last layer is printed, showing which neurons fired.
5) **Memory Cleanup**: The dynamically allocated memory for the network is freed.

III. Classification Model

The classification model is implemented in `classify_snn.py`. The model is a small SNN that classifies data with two classes. The data is a single float value per sample, and it is converted to spikes for the classification task. The model consists of two Leaky Integrate-and-Fire (LIF) layers, each with 10 neurons. The training loop involves converting the data to spike trains, computing the loss using cross-entropy, and updating the model parameters using backpropagation.

IV. Data

The data used in this project is generated using `generate_dummy_data.py`. The data consists of two classes sampled from normal distributions. Class 1 is sampled from a normal distribution with mean 1 and standard deviation 1, while Class 2 is sampled from a normal distribution with mean -1 and standard deviation 1. The data and labels are saved to text files for use in the simulation and classification tasks.

V. Conclusion

This project demonstrates the simulation of an SNN and the implementation of a small SNN for data classification. The main simulation code processes input spikes through the network layers and updates the neurons' states based on the input and their weights. The classification model converts data to spikes and classifies it using a small SNN. The data used in this project is generated from normal distributions and saved to text files. This project provides a comprehensive understanding of SNNs and their applications in simulation and classification tasks.

Acknowledgment

The authors would like to thank the Department of Electrical and Computer Engineering at Drexel University for their support and resources.