A Scatter-and-Gather Spiking Convolutional Neural Network on a Reconfigurable Neuromorphic Hardware

---

This code can be used as supplemental material for three papers:

- Principle of spatio-temporal ANN-to-SNN conversion:
  - "A Novel Conversion Method for Spiking Neural Network using Median Quantization", *IEEE ISCAS*, October, 2020.
- Spatial conversion and mapping on PAICore2.0 (Scatter-and-Gather, SG):
  - "A Scatter-and-Gather Spiking Convolutional Neural Network on a Reconfigurable Neuromorphic Hardware", *Frontiers in Neuroscience*, October, 2021.
- Temporal conversion and mapping on PAICore1.0 (Store-and-Release, SR):
  - "Modular Building Blocks for Mapping Spiking Neural Networks onto a Programmable Neuromorphic Processor", *Elsevier Microelectronics Journal*, revised, October, 2022.

Citation:

To be completed.

Features:

- This supplemental material gives a reproduction function of ANN training, testing and converted SNN inference experiments in our paper. Besides, visualized results for spiking sparsity and synaptic operations (SOPs) are provided.

File overview:

- `README.md` - this readme file.
- `video_for_demonstration.webm` - a video for demonstration using PAICore1.0 (PKU-NC64C).
- `LeNet` - the project folder for LeNet.
- `VGG`- the project folder for VGG-Net.

Requirements

Dependencies and Libraries:

- python 3.5 (https://www.python.org/ or https://www.anaconda.com/)
- tensorflow_gpu 1.2.1 (https://github.com/tensorflow)
- tensorlayer 1.8.5 (https://github.com/tensorlayer)
- CPU: Intel(R) Xeon(R) CPU E5-2620 v4 @ 2.10GHz
- GPU: Tesla V100

Installation:

To install requirements,

```
pip install -r requirements.txt
```

Datasets:

- MNIST: dataset, preprocessing
- CIFAR10/100: dataset, preprocessing

ANN Training

Before running:

- Please installing the required package Tensorflow and Tensorlayer (using our modified version)
- Please note your default dataset folder will be `workspace/data`, such as `Spatio_temporal_SNNs/LeNet/data`
- Select the index of GPU in the training scripts (0 by default)

Run the code:

for example (ANN training, *k=0*, *B=1*, LeNet, MNIST):

```
$ cd LeNet
$ python Quant_LeNet_MNIST.py  --k 0 --B 1 --resume False --learning_rate 0.001 --
mode 'training'
```

ANN Inference

Run the code:

for example (ANN inference, *k=0*, *B=1*, LeNet, MNIST):

```
$ python Quant_LeNet_MNIST.py  --k 0 --B 1 --resume True --mode 'inference'
```

SNN inference

Run the code:

for example (SNN inference, *k=0*, *B=1*, LeNet, MNIST):

```
$ python Spiking_LeNet_MNIST.py  --k 0 --B 1 --noise_ratio 0
```

it will generate the corresponding log files including: `accuracy.txt`, `sop_num.txt`, `spike_collect.txt` and `spike_num.txt` in `./figs/k0B1`.

Others

- We do not consider the synaptic operations in the input encoding layer and the spike output in the last classification layer (membrane potential accumulation) for both original ANN counterparts and converted SNNs.
- More instructions for running the code can be found in the respective workspace folder (`LeNet/README_LeNet.md`, `VGG/README_VGG.md`).

Results

Our proposed methods achieve the following performances on MNIST, CIFAR10/100:

MNIST:

Quantization Precision	Network Size	Epochs	ANN	SNN	Time Steps
Full-precision	16C5-P2-16C5-P2-256	200	99.52%	N/A	N/A
k=0, B=1	16C5-P2-16C5-P2-256	200	99.27%	99.27%	1
k=0, B=2	16C5-P2-16C5-P2-256	200	99.32%	99.32%	1
k=0, B=4	16C5-P2-16C5-P2-256	200	99.43%	99.43%	1
k=1, B=1	16C5-P2-16C5-P2-256	200	99.30%	99.30%	1
k=1, B=2	16C5-P2-16C5-P2-256	200	99.37%	99.37%	1
k=1, B=4	16C5-P2-16C5-P2-256	200	99.50%	99.50%	1

CIFAR10:

Quantization Level	Network Size	Epochs	ANN	SNN	Time Steps
full-precision	64C3*2-2P2-128C3*2-P2-256C3*2-P2-512C3-512	400	92.85%	N/A	N/A
k=0, B=1	64C3*2-2P2-128C3*2-P2-256C3*2-P2-512C3-512	400	89.12%	89.12%	1
k=0, B=2	64C3*2-2P2-128C3*2-P2-256C3*2-P2-512C3-512	400	90.95%	90.95%	1
k=0, B=4	64C3*2-2P2-128C3*2-P2-256C3*2-P2-512C3-512	400	91.65%	91.65%	1
k=1, B=1	64C3*2-2P2-128C3*2-P2-256C3*2-P2-512C3-512	400	90.14%	90.14%	1
k=1, B=2	64C3*2-2P2-128C3*2-P2-256C3*2-P2-512C3-512	400	91.91%	91.91%	1

Quantization Level	Network Size	Epochs	ANN	SNN	Time Steps
k=1, B=4	64C3*2-2P2-128C3*2-P2-256C3*2-P2-512C3-512	400	92.27%	92.27%	1

CIFAR100:

Quantization Level	Network Size	Epochs	ANN	SNN	Time Steps
full-precision	64C3*2-2P2-128C3*2-P2-256C3*2-P2-512C3-512	400	67.4%	N/A	N/A
k=0, B=1	64C3*2-2P2-128C3*2-P2-256C3*2-P2-512C3-512	400	56.1%	56.1%	1
k=0, B=2	64C3*2-2P2-128C3*2-P2-256C3*2-P2-512C3-512	400	62.5%	62.5%	1
k=0, B=4	64C3*2-2P2-128C3*2-P2-256C3*2-P2-512C3-512	400	65.6%	65.6%	1
k=1, B=1	64C3*2-2P2-128C3*2-P2-256C3*2-P2-512C3-512	400	59.2%	59.2%	1
k=1, B=2	64C3*2-2P2-128C3*2-P2-256C3*2-P2-512C3-512	400	65.0%	65.0%	1
k=1, B=4	64C3*2-2P2-128C3*2-P2-256C3*2-P2-512C3-512	400	66.2%	66.2%	1

More question:

- There might be a little difference of results for multiple training repetitions, because of the randomization.
- Please feel free to reach out here or email: 1801111301@pku.edu.cn, if you have any questions or difficulties. I'm happy to help guide you.