Instructions for running CIFAR10 experiments

File overview:

- `README_VGG.md` - this readme file for VGG-Net on CIFAR10/100.

- `CIFAR10` - VGG-Net on CIFAR10.

    - `tensorlayer` - our provided tensorlayer package.
    - `Quant_VGG_CIFAR10.py` - the training script for `VGG-Net` with optional quantization precision k on CIFAR10.
    - `Spiking_VGG_CIFAR10.py` - the evaluation script for `spiking VGG-Net` with optional quantization precision k on CIFAR10.
    - `FP32_VGG_CIFAR10.py` - the training script for `VGG-Net` with `full precision (float32)` on CIFAR10.
    - `spiking_ulils.py` - the functions of spiking convolution and linear.
    - `figs` - visualization folder for SNN performance.
        - `sops.py` - the synaptic operations (SOPs) script for `spiking VGG-Net` with different quantization precisions on CIFAR10.
        - `sparsity.py` - the spike sparsity script for `spiking VGG-Net` with different quantization precisions on CIFAR10.

- `CIFAR100` - VGG-Net on CIFAR100.

    - `tensorlayer` - our provided tensorlayer package.
    - `Quant_VGG_CIFAR100.py` - the training script for `VGG-Net` with optional quantization precision k on CIFAR100.
    - `Spiking_VGG_CIFAR100.py` - the evaluation script for `spiking VGG-Net` with optional quantization precision k on CIFAR100.
    - `FP32_VGG_CIFAR100.py` - the training script for `VGG-Net` with `full precision (float32)` on CIFAR100.
    - `spiking_ulils.py` - the functions of spiking convolution and linear.
    - `figs` - visualization folder for SNN performance.
        - `sops.py` - the synaptic operations (SOPs) script for `spiking VGG-Net` with different quantization precisions on CIFAR100.
        - `sparsity.py` - the spike sparsity script for `spiking VGG-Net` with different quantization precisions on CIFAR100.

ANN Training

Before running:

- Please note your default dataset folder will be `./data`

- please modify the command line parameters: `--resume True`, and `--learning_rate 0.0001` for another 200 epochs after the first 200 epochs. Totally, 400 training epoch need be performed.

Run the code:

for example (training, *k=0*, *B=1*, VGG-Net, CIFAR10):

```
$ cd VGG/CIFAR10
$ python Quant_VGG_CIFAR10.py  --k 0 --B 1--resume False --learning_rate 0.001 --
mode 'training'
```

finally, it will generate the corresponding model files including: `checkpoint`, `model_CIFAR10_advanced.ckpt.data-00000-of-00001`, `model_CIFAR10_advanced.ckpt.index`, `model_CIFAR10_advanced.ckpt.meta` and `model_cifar_10.npz`.

ANN Inference

Run the code:

for example (inference, *k=0*, *B=1* VGG-Net, CIFAR10):

```
$ python Quant_VGG_CIFAR10.py --k 0 --B 1--resume True  --mode 'inference'
```

Then, it will print the corresponding ANN test accuracy.

SNN inference

Run the code:

for example (inference, *k=0*, *B=1*, VGG-Net, CIFAR10):

```
$ python Spiking_VGG_CIFAR10.py --k 0 --B 1 --noise_ratio 0
```

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

Visualization

Firing sparsity:

```
$ cd figs
$ python sparsity.py
```

Computing operations:

```
$ python sops.py
```

Results

Our proposed spiking VGG-Net achieves the following performances on CIFAR10/100:

VGG-Net: 64C3*2-2P2-128C3*2-P2-256C3*2-P2-512C3-512

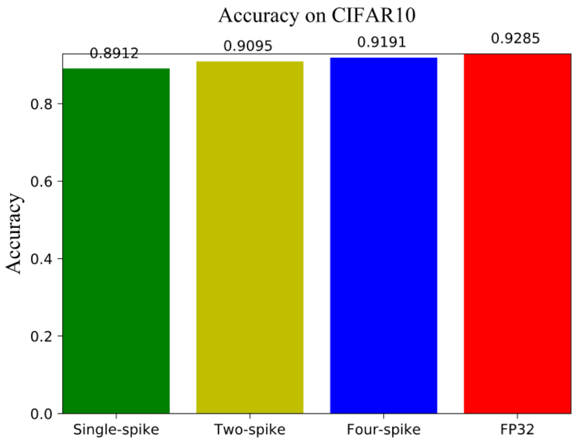
Accuracy:

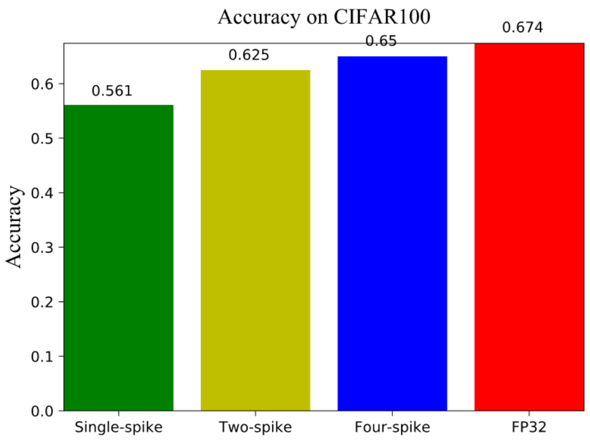
CIFAR10:

Quantization Precision	Network	Epochs	ANN	SNN	Time Steps
Full-precision	VGG-Net	400	92.85%	N/A	N/A
k=1, B=2	VGG-Net	400	91.91%	91.91%	1
k=1, B=2 (10% noise)	VGG-Net	400	91.91%	90.32%%	1
k=1, B=2 (20% noise)	VGG-Net	400	91.91%	89.65%	1

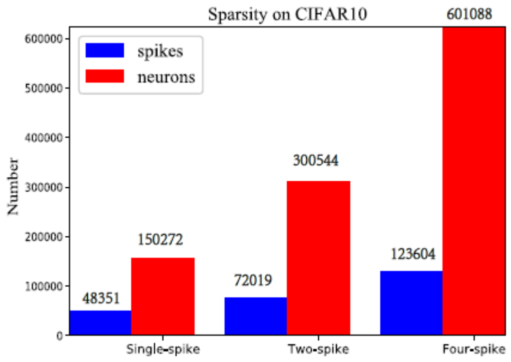
CIFAR100:

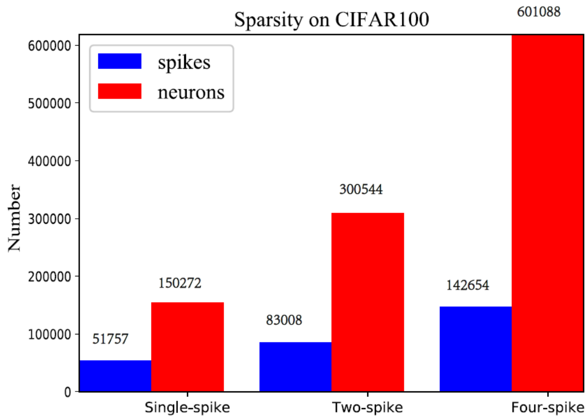
Quantization Precision	Network	Epochs	ANN	SNN	Time Steps
Full-precision	VGG-Net	400	67.4%	N/A	N/A
k=1, B=2	VGG-Net	400	65.0%	65.0%	1
k=1, B=2 (10% noise)	VGG-Net	400	65.0%	63.93%	1
k=1, B=2 (20% noise)	VGG-Net	400	65.0%	62.25%	1



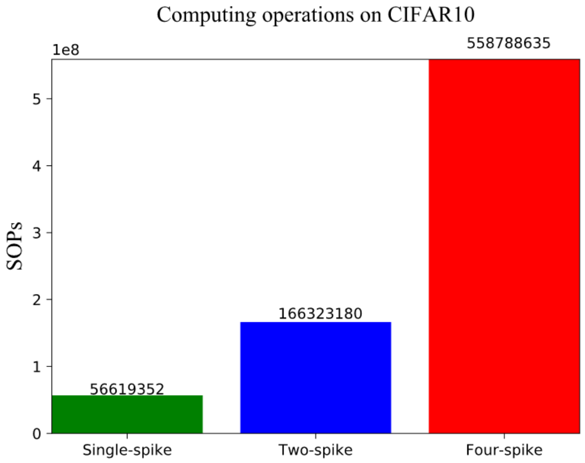


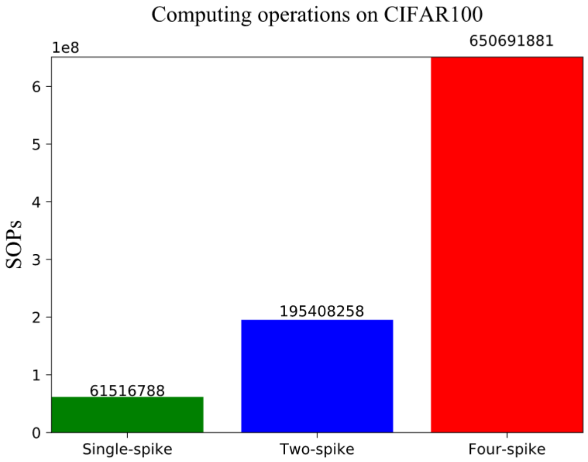
Firing sparsity:





Computing operations:





Notes

- We do not consider the synaptic operations in the input encoding layer and the spike outputs in the last classification layer (membrane potential accumulation instead) for both original ANN counterparts and converted SNNs.
- We also provide some scripts for visualization in `./figs`, please move to this folder and directly run the three scripts.

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.