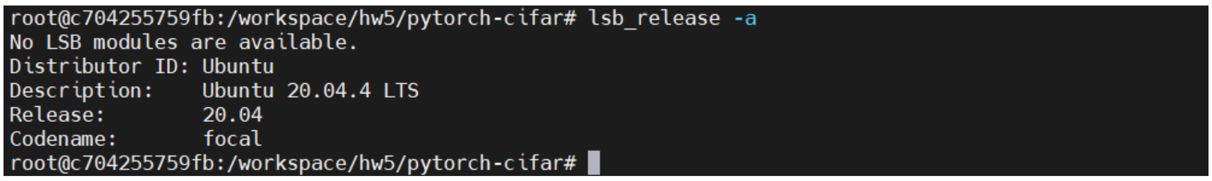
Student ID: 410551027

Student name: 江衍涵

1. I am using a high-speed computing computer, system Information as follows:

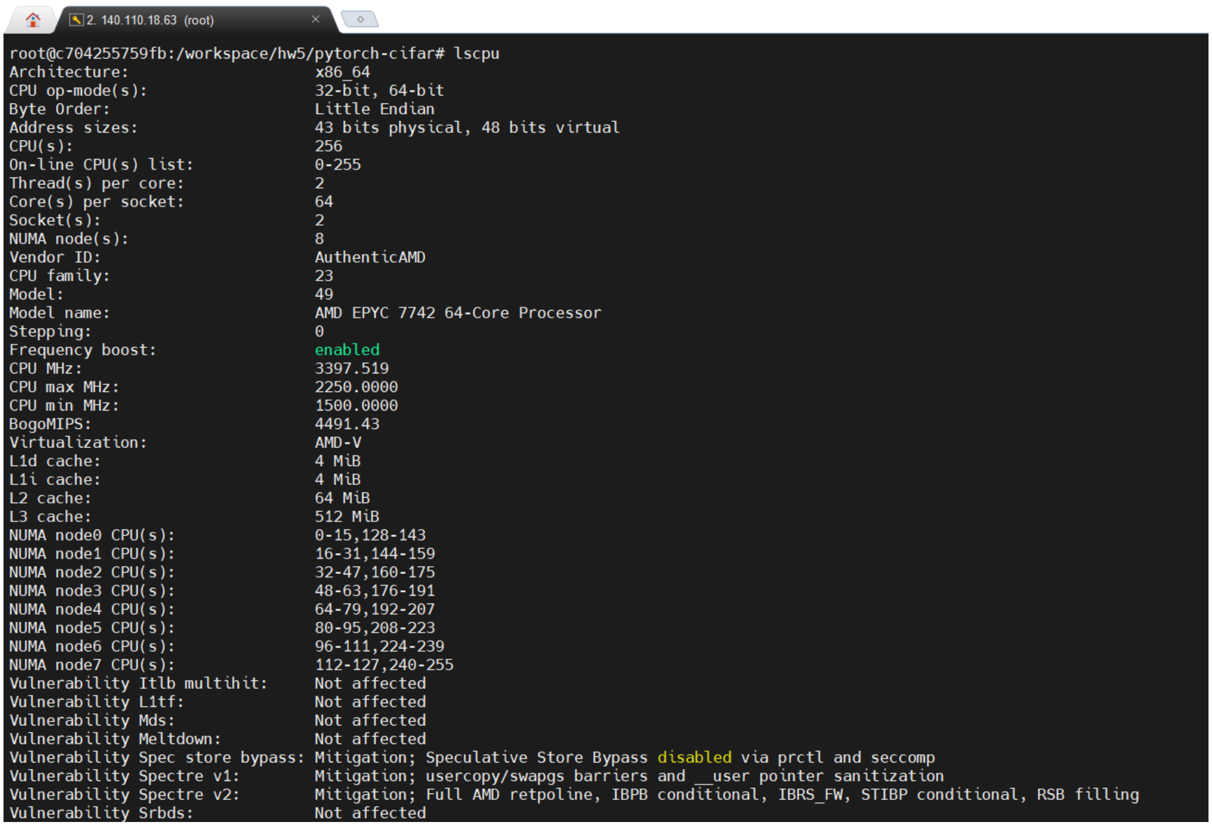
Hardware details

OS info:



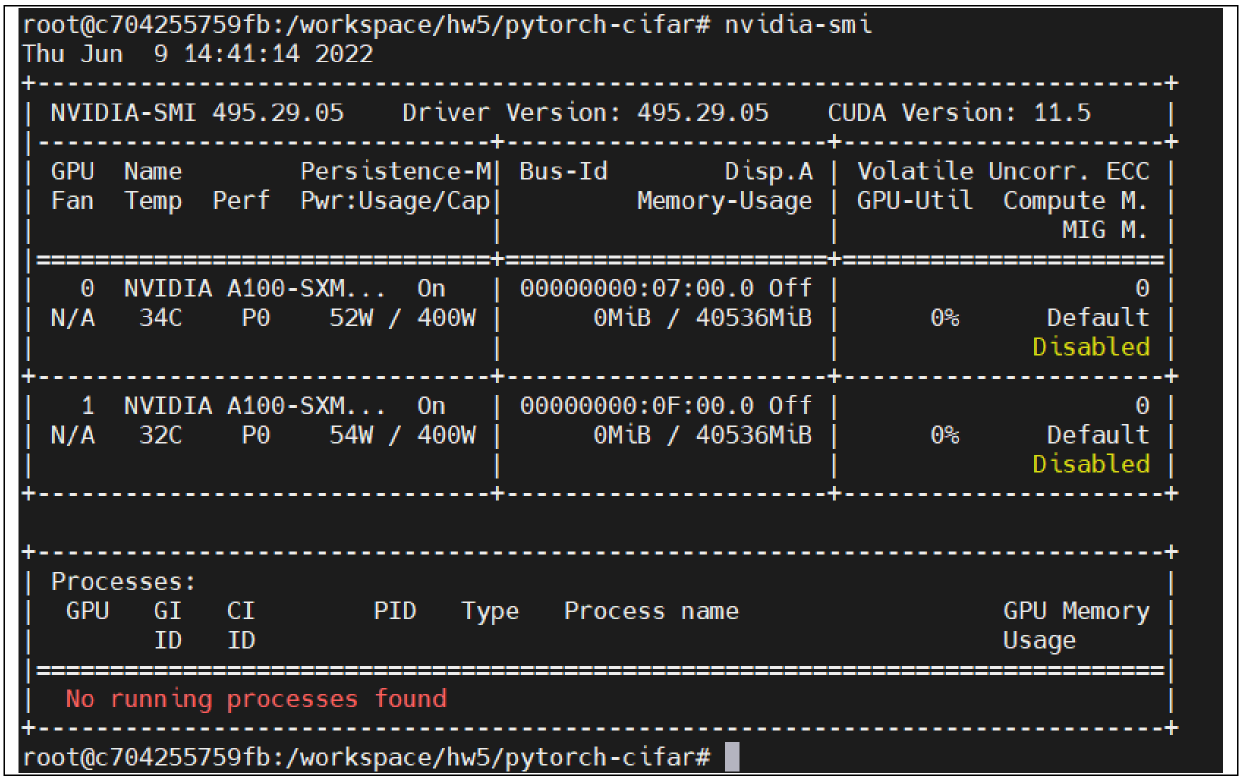
```
root@c704255759fb:/workspace/hw5/pytorch-cifar# lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 20.04.4 LTS
Release:        20.04
Codename:       focal
root@c704255759fb:/workspace/hw5/pytorch-cifar#
```

CPU info:



```
root@c704255759fb:/workspace/hw5/pytorch-cifar# lscpu
Architecture:                    x86_64
CPU op-mode(s):                  32-bit, 64-bit
Byte Order:                      Little Endian
Address sizes:                   43 bits physical, 48 bits virtual
CPU(s):                          256
On-line CPU(s) list:             0-255
Thread(s) per core:              2
Core(s) per socket:              64
Socket(s):                       2
NUMA node(s):                    8
Vendor ID:                       AuthenticAMD
CPU family:                      23
Model:                           49
Model name:                      AMD EPYC 7742 64-Core Processor
Stepping:                        0
Frequency boost:                 enabled
CPU MHz:                         3397.519
CPU max MHz:                     2250.0000
CPU min MHz:                     1500.0000
BogoMIPS:                        4491.43
Virtualization:                  AMD-V
L1d cache:                       4 MiB
L1i cache:                       4 MiB
L2 cache:                        64 MiB
L3 cache:                        512 MiB
NUMA node0 CPU(s):               0-15,128-143
NUMA node1 CPU(s):               16-31,144-159
NUMA node2 CPU(s):               32-47,160-175
NUMA node3 CPU(s):               48-63,176-191
NUMA node4 CPU(s):               64-79,192-207
NUMA node5 CPU(s):               80-95,208-223
NUMA node6 CPU(s):               96-111,224-239
NUMA node7 CPU(s):               112-127,240-255
Vulnerability Itlb multihit:     Not affected
Vulnerability L1tf:              Not affected
Vulnerability Mds:               Not affected
Vulnerability Meltdown:          Not affected
Vulnerability Spec store bypass: Mitigation; Speculative Store Bypass disabled via prctl and seccomp
Vulnerability Spectre v1:        Mitigation; usercopy/swapgs barriers and __user pointer sanitization
Vulnerability Spectre v2:        Mitigation; Full AMD retpoline, IBPB conditional, IBRS_FW, STIBP conditional, RSB filling
Vulnerability Srbds:             Not affected
```

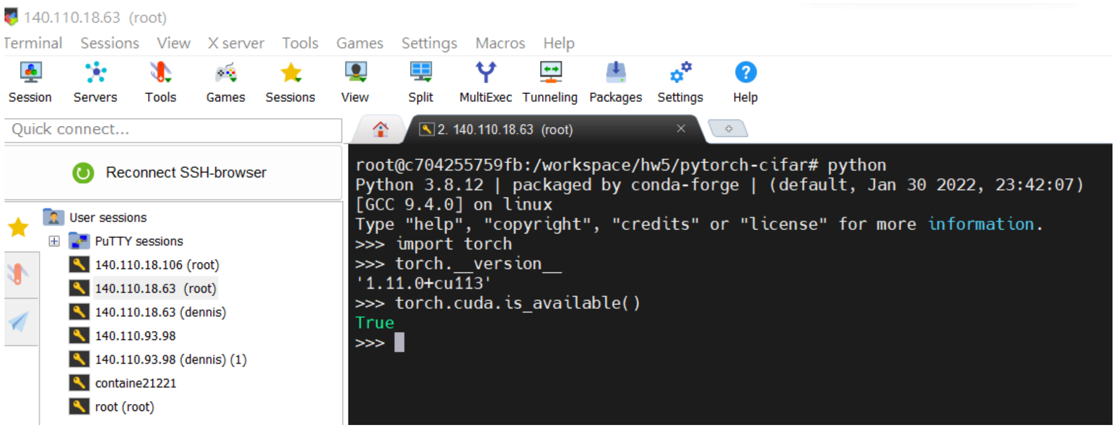
GPU info:



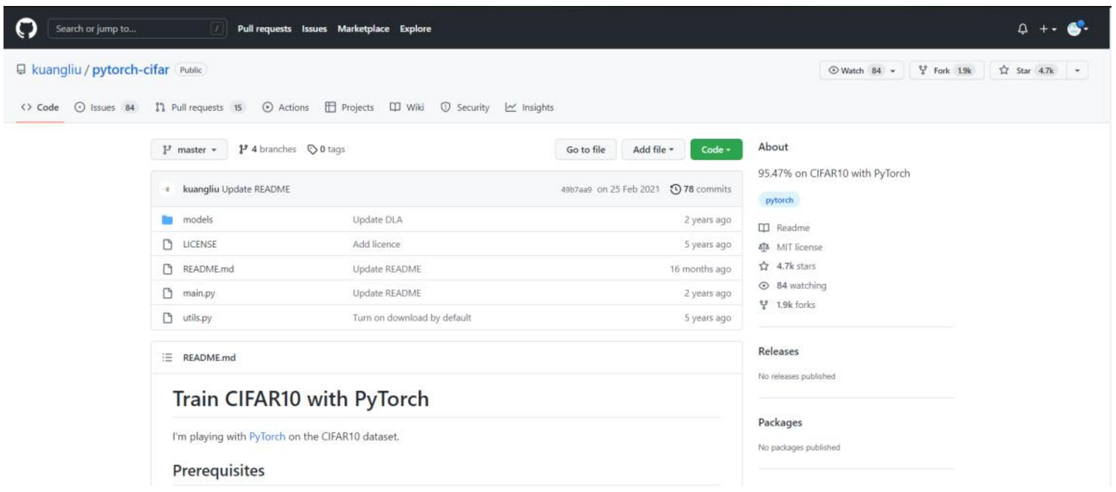
```
root@c704255759fb:/workspace/hw5/pytorch-cifar# nvidia-smi
Thu Jun  9 14:41:14 2022
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 495.29.05    Driver Version: 495.29.05    CUDA Version: 11.5     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  NVIDIA A100-SXM...  On   | 00000000:07:00.0 Off |                    0 |
| N/A   34C    P0    52W / 400W |      0MiB / 40536MiB |      0%      Default |
|                               |                      |             Disabled |
+-------------------------------+----------------------+----------------------+
|   1  NVIDIA A100-SXM...  On   | 00000000:0F:00.0 Off |                    0 |
| N/A   32C    P0    54W / 400W |      0MiB / 40536MiB |      0%      Default |
|                               |                      |             Disabled |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
root@c704255759fb:/workspace/hw5/pytorch-cifar#
```

2. I am using pytorch-cifar(repository) on github, which is a paper called "Deep Layer Aggregation"published in the CVPR journal. I use this repository to write homework 5 assigments. It's deep learning framework is pytorch, and version is as follows. My environment is running in Docker, and its tag is "cyh1195/cyh:1110612cs_hw5". The file path is in "/workspace/code".

- Python 3.8.12
- PyTorch 1.11+cu113
- https://github.com/kuangliu/pytorch-cifar
- docker tag: docker pull cyh1195/cyh:1110612cs_hw5



```
root@c704255759fb:/workspace/hw5/pytorch-cifar# python
Python 3.8.12 | packaged by conda-forge | (default, Jan 30 2022, 23:42:07)
[GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import torch
>>> torch.__version__
'1.11.0+cu113'
>>> torch.cuda.is_available()
True
>>>
```



kuangliu / pytorch-cifar Public

4 branches 0 tags

kuangliu Update README 49b7aa9 on 25 Feb 2021 78 commits

models	Update DLA	2 years ago
LICENSE	Add licence	5 years ago
README.md	Update README	16 months ago
main.py	Update README	2 years ago
utils.py	Turn on download by default	5 years ago

README.md

Train CIFAR10 with PyTorch

I'm playing with PyTorch on the CIFAR10 dataset.

Prerequisites

About

95.47% on CIFAR10 with PyTorch

pytorch

Readme
MIT license
4.7k stars
84 watching
1.9k forks

Releases

No releases published

Packages

No packages published

3. How to use the program?

```
# Start training with:
python train.py

# You can manually resume the training with:
python train.py –resume

# Using Model Inference:
python inference.py
```

4. Model desinged:

I use the model structure of this paper, and his model hyperparameter settings as in the Table 1. Data pre-processing is done only for normalization, and no other actions are done. This deep layer aggregation structures iteratively and hierarchically merge the feature hierarchy to make networks with better accuracy and fewer parameters. Aggregation is a decisive aspect of architecture as Fig 1, and as the number of modules multiply their connectivity is made all the more important. By relating architectures for aggregating channels, scales, and resolutions we identified the need for deeper aggregation, and addressed it by iterative deep aggregation and hierarchical deep aggregation.

Table 1. Hyperparameter settings

model architecture	SimpleDLA
optimizer	SGD
learning rate	0.0001
momentum	0.9
weight_decay	0.0005
batch size	256
epoch	5000
criterion(loss function)	CrossEntropyLoss
scheduler	CosineAnnealingLR

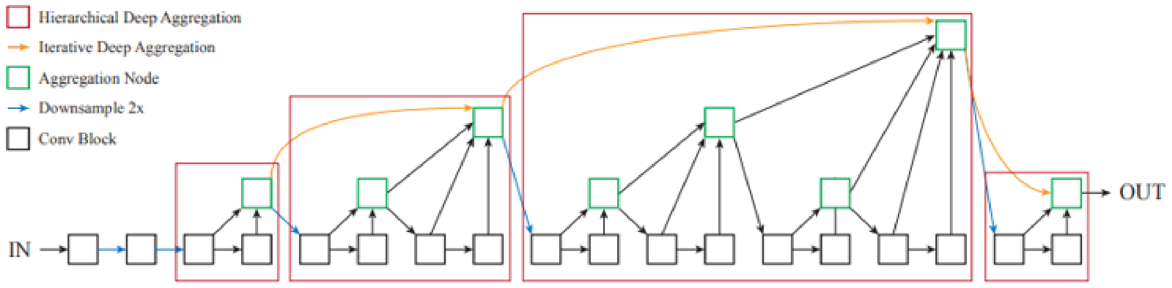
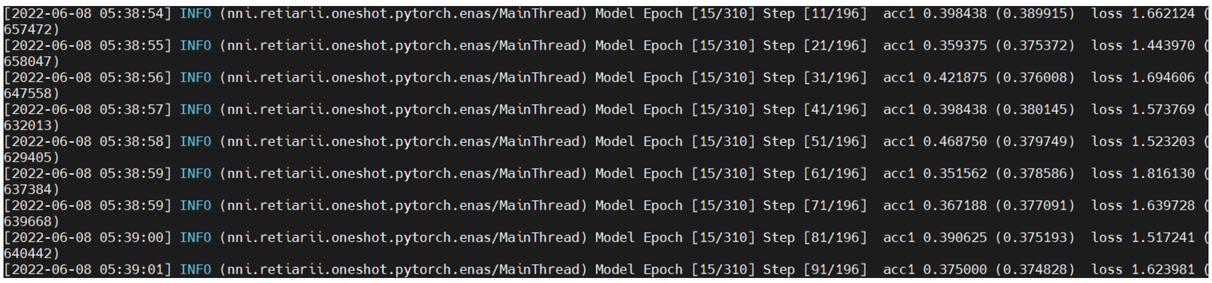


Figure 1: Deep layer aggregation

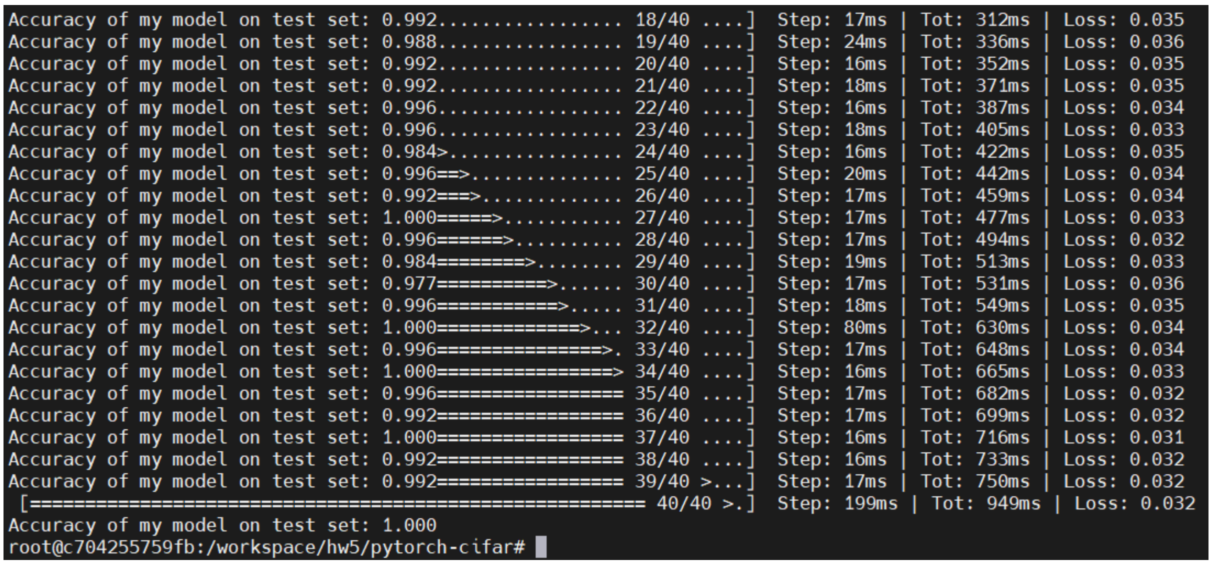
5. Result:

First, I trained the SimpleDLA model 195 times using training.py and the original cifar-10, the training process is shown in Figure 2. After training, the weights are saved in the "ckpt_1000.pth" file as a pre-trained model. Then use the x_train.npy dataset from Homework 5, and train it again with the same hyperparameter settings. Finally, the test.npy dataset is tested with inference.py, and the average accuracy of the test.npy dataset is 99%.



```
[2022-06-08 05:38:54] INFO (nni.retiarii.oneshot.pytorch.enas/MainThread) Model Epoch [15/310] Step [11/196]  acc1 0.398438 (0.389915)  loss 1.662124 (
657472)
[2022-06-08 05:38:55] INFO (nni.retiarii.oneshot.pytorch.enas/MainThread) Model Epoch [15/310] Step [21/196]  acc1 0.359375 (0.375372)  loss 1.443970 (
658047)
[2022-06-08 05:38:56] INFO (nni.retiarii.oneshot.pytorch.enas/MainThread) Model Epoch [15/310] Step [31/196]  acc1 0.421875 (0.376008)  loss 1.694606 (
647558)
[2022-06-08 05:38:57] INFO (nni.retiarii.oneshot.pytorch.enas/MainThread) Model Epoch [15/310] Step [41/196]  acc1 0.398438 (0.380145)  loss 1.573769 (
632013)
[2022-06-08 05:38:58] INFO (nni.retiarii.oneshot.pytorch.enas/MainThread) Model Epoch [15/310] Step [51/196]  acc1 0.468750 (0.379749)  loss 1.523203 (
629405)
[2022-06-08 05:38:59] INFO (nni.retiarii.oneshot.pytorch.enas/MainThread) Model Epoch [15/310] Step [61/196]  acc1 0.351562 (0.378586)  loss 1.816130 (
637384)
[2022-06-08 05:38:59] INFO (nni.retiarii.oneshot.pytorch.enas/MainThread) Model Epoch [15/310] Step [71/196]  acc1 0.367188 (0.377091)  loss 1.639728 (
639668)
[2022-06-08 05:39:00] INFO (nni.retiarii.oneshot.pytorch.enas/MainThread) Model Epoch [15/310] Step [81/196]  acc1 0.390625 (0.375193)  loss 1.517241 (
640442)
[2022-06-08 05:39:01] INFO (nni.retiarii.oneshot.pytorch.enas/MainThread) Model Epoch [15/310] Step [91/196]  acc1 0.375000 (0.374828)  loss 1.623981 (
```

Figure 2: History of traing model



```
Accuracy of my model on test set: 0.992................. 18/40 ....]  Step: 17ms | Tot: 312ms | Loss: 0.035
Accuracy of my model on test set: 0.988................. 19/40 ....]  Step: 24ms | Tot: 336ms | Loss: 0.036
Accuracy of my model on test set: 0.992................. 20/40 ....]  Step: 16ms | Tot: 352ms | Loss: 0.035
Accuracy of my model on test set: 0.992................. 21/40 ....]  Step: 18ms | Tot: 371ms | Loss: 0.035
Accuracy of my model on test set: 0.996................. 22/40 ....]  Step: 16ms | Tot: 387ms | Loss: 0.034
Accuracy of my model on test set: 0.996................. 23/40 ....]  Step: 18ms | Tot: 405ms | Loss: 0.033
Accuracy of my model on test set: 0.984>................ 24/40 ....]  Step: 16ms | Tot: 422ms | Loss: 0.035
Accuracy of my model on test set: 0.996==>.............. 25/40 ....]  Step: 20ms | Tot: 442ms | Loss: 0.034
Accuracy of my model on test set: 0.992===>............. 26/40 ....]  Step: 17ms | Tot: 459ms | Loss: 0.034
Accuracy of my model on test set: 1.000=====>........... 27/40 ....]  Step: 17ms | Tot: 477ms | Loss: 0.033
Accuracy of my model on test set: 0.996======>.......... 28/40 ....]  Step: 17ms | Tot: 494ms | Loss: 0.032
Accuracy of my model on test set: 0.984========>........ 29/40 ....]  Step: 19ms | Tot: 513ms | Loss: 0.033
Accuracy of my model on test set: 0.977==========>...... 30/40 ....]  Step: 17ms | Tot: 531ms | Loss: 0.036
Accuracy of my model on test set: 0.996===========>..... 31/40 ....]  Step: 18ms | Tot: 549ms | Loss: 0.035
Accuracy of my model on test set: 1.000=============>... 32/40 ....]  Step: 80ms | Tot: 630ms | Loss: 0.034
Accuracy of my model on test set: 0.996===============>. 33/40 ....]  Step: 17ms | Tot: 648ms | Loss: 0.034
Accuracy of my model on test set: 1.000================> 34/40 ....]  Step: 16ms | Tot: 665ms | Loss: 0.033
Accuracy of my model on test set: 0.996================= 35/40 ....]  Step: 17ms | Tot: 682ms | Loss: 0.032
Accuracy of my model on test set: 0.992================= 36/40 ....]  Step: 17ms | Tot: 699ms | Loss: 0.032
Accuracy of my model on test set: 1.000================= 37/40 ....]  Step: 16ms | Tot: 716ms | Loss: 0.031
Accuracy of my model on test set: 0.992================= 38/40 ....]  Step: 16ms | Tot: 733ms | Loss: 0.032
Accuracy of my model on test set: 0.992================= 39/40 >...]  Step: 17ms | Tot: 750ms | Loss: 0.032
 [====================================================== 40/40 >.]  Step: 199ms | Tot: 949ms | Loss: 0.032
Accuracy of my model on test set: 1.000
root@c704255759fb:/workspace/hw5/pytorch-cifar# 
```

Figure 3: The inferend result from testing data is an accuracy of 99%