PT_SOLVER speedup with Deep Neural Network

Xingqiu Yuan

Princeton Plasma Physics Lab

Presented at TUG transp Users meeting on May 5th 2017

current progress on DNN

WHITE PAPER

Information Technology
Machine Learning Solutions

intel

Caffe* Optimized for Intel® Architecture: Applying Modern Code Techniques

Improving the computational performance of a deep learning framework

Authors

Vadim Karpusenko, Ph.D.
Intel Corporation

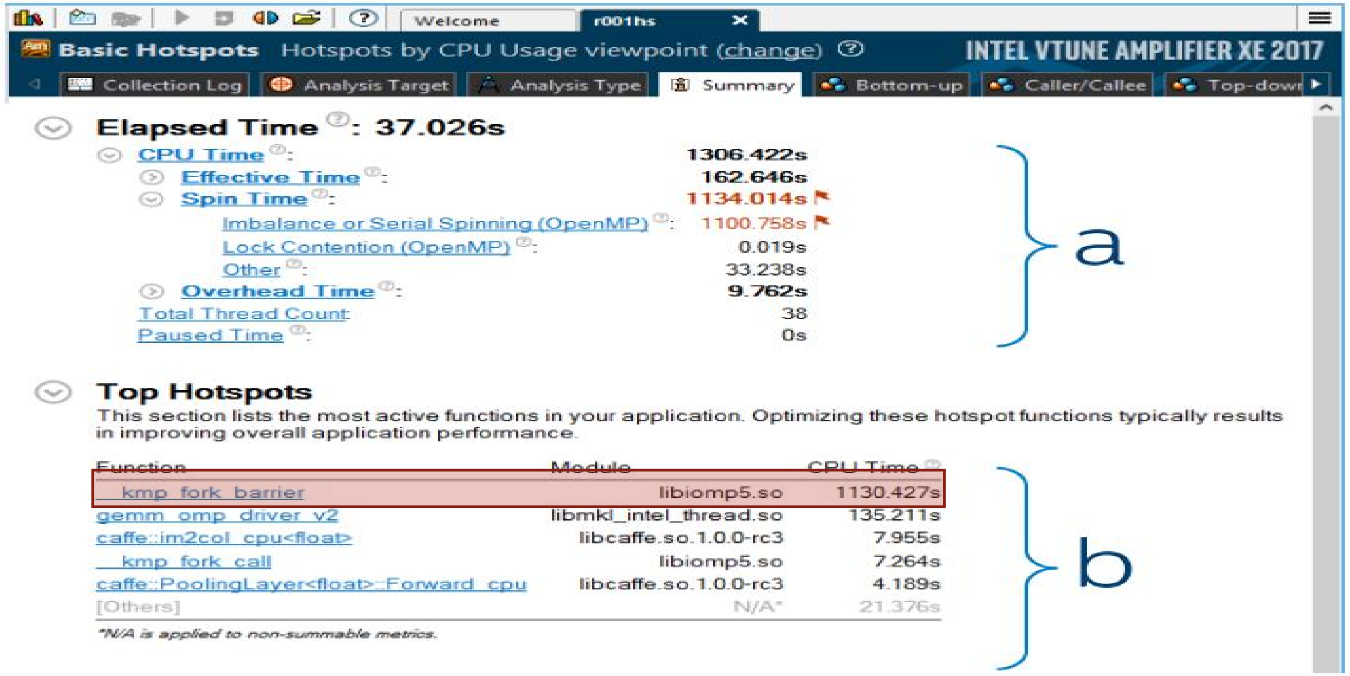
Andres Rodriguez, Ph.D.
Intel Corporation

Jacek Czaja
Intel Corporation

Mariusz Moczala
Intel Corporation

Abstract

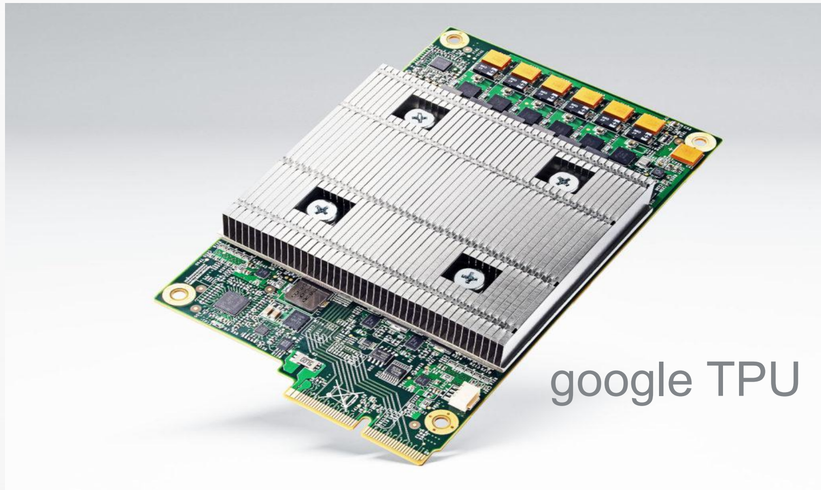
This paper demonstrates a special version of Caffe*—a deep learning framework originally developed by the Berkeley Vision and Learning Center (BVLC)—that is optimized for Intel® architecture. This version of Caffe, known as Caffe optimized for Intel architecture, is currently integrated with the latest release of Intel® Math Kernel Library 2017 and is optimized for Intel® Advanced Vector Extensions 2 and will include Intel Advanced Vector Extensions 512 instructions. This solution is supported by Intel® Xeon® processors and Intel® Xeon Phi™ processors, among others. This paper includes performance results for a CIFAR-10* image-classification dataset, and it describes the tools and code modifications that can be used to improve computational performance for the BVLC Caffe code and other deep learning frameworks.



News room > News releases >

IBM Linux Servers Designed to Accelerate Artificial Intelligence, Deep Learning and Advanced Analytics

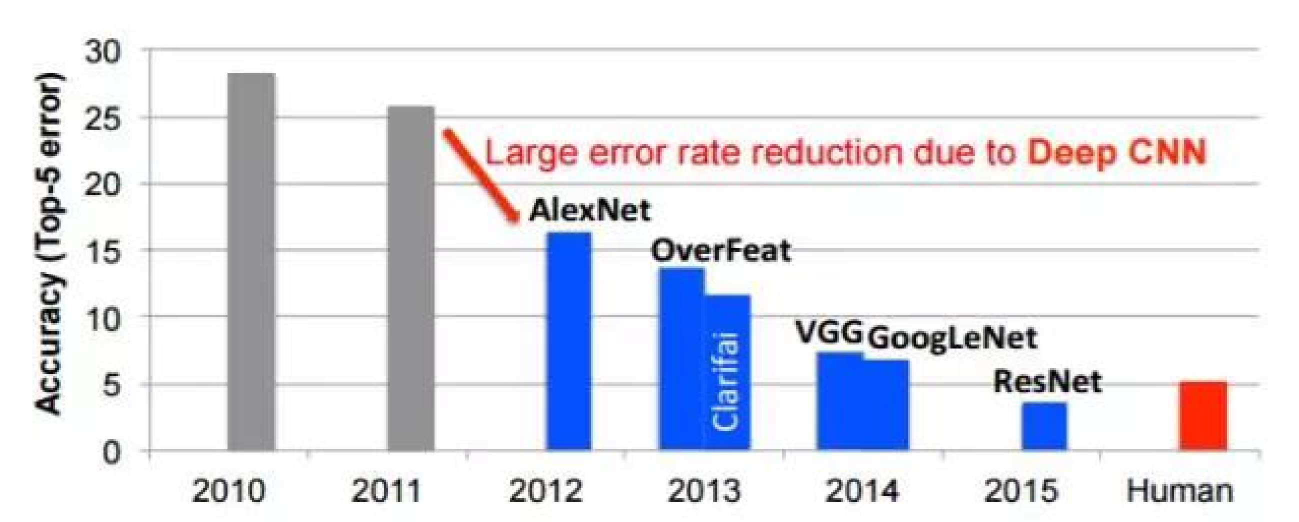
- New IBM POWER8 Chip with NVIDIA NVLink(TM) Enables Data Movement 5x Faster than Any Competing Platform
- Systems Deliver Average of 80% More Performance Per Dollar than Latest x86-Based Servers(1)
- Expanded Linux Server Lineup Leverages OpenPOWER Innovations



TensorFlow

Many super company made big progress in speed up deep-learning package, include Nvidia, Cray, IBM, BAIDU, and INTEL

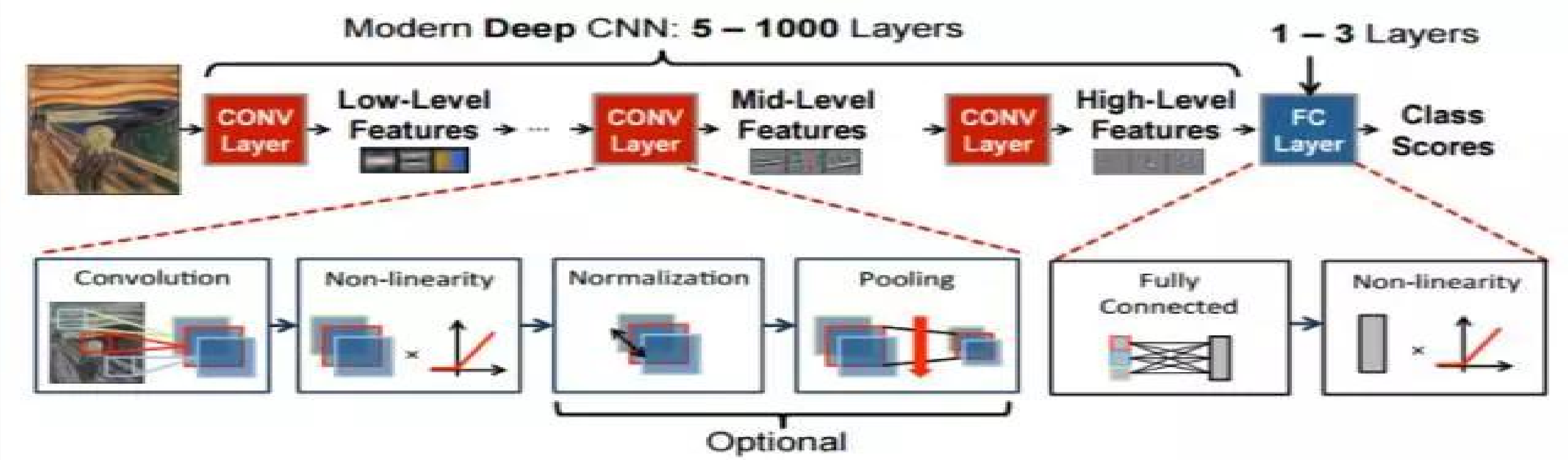
Deep Neural Network is accurate but often overfiting



IMAGENET
Large Scale Visual Recognition Challenge (ILSVRC)
http://image-net.org/challenges/LSVRC/

using very deep neural network, DNN provides accuracy better than humain beging.

more than 1000 layers achieved.



current status of PT_SOLVER

- Multilevel parallelization (over ky inside TGLF, and flux surface), no limitation on the number of flux surface, upto more than 1k cores can be used on NERSC.
- TGLF is slow, and gives discontinuous fluxes, not good for convergence. (over a week to finish ITER 400s discharge case)
- Many runs have been performed for different devices, and different parameter regions (easy to establish database)

- Feedforward propagation deep learning neural network is very fast and accurate as an alternative of TGLF to provide turbulent fluxes.(10 to 100 times faster than TGLF)
- Continuous fluxes function from DNN, better convergence.
- However, it is hard to train a deep neural network (overfiting, gradient vanishing, local minimum, massively parallelization, big database, etc).

current implementation of DNN package

- many of the open source packages allow CPU/GPU hybrid computing
- but using offload mode, the optimization work was given for GPU for efficient matrix operation (vectorization or data parallelization).
- the foundmental class is layer (in paddle-paddle, caffe, and so on), expcept tensor flow.
- the levels of parallelism are limited (scalability?)

- however, the current exascale computing requires many levels of parallelization, and many runtime concurrency.
- actor based programming paradigm gives many levels of parallelism, and suitable for hybrid computing.

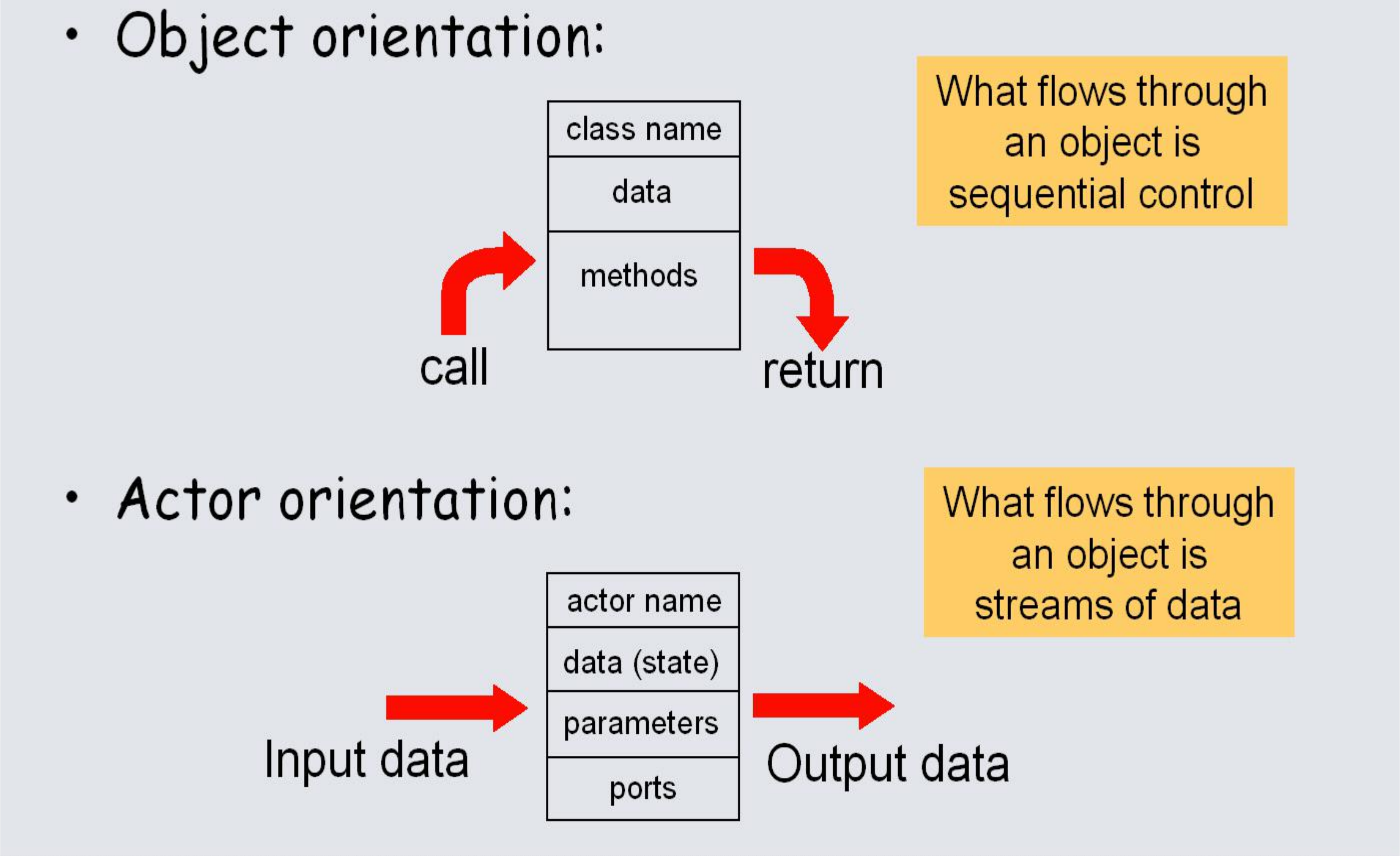
Parallel Programming Paradigms for Heterogeneous Architecture

- 1): asynchronous programming
  a): global synchronization is expensive and waste of cpu time
  b): local synchronization is allowed.

- 2): hybrid task & data parallelism
  a): OpenMP, TBB, Cilk and OpenACC etc are only allow data parallelism (vectorization)
  b): task parallelism is difficult

- 3): actor based programming paradigm
  a): actors are reactive object, dynamically created/destroied
  b): actors execute concurrently
  c): actors are small tasks that are scalable.
  d): examples, such as HPX, intel OCR, Legion, CAF, and Kepler

Actors are similar to c++ objects, but more...



Actor-Dataflow Orientation vs Object-Control flow Orientation

actor based programming paradigm (with application to DNN)

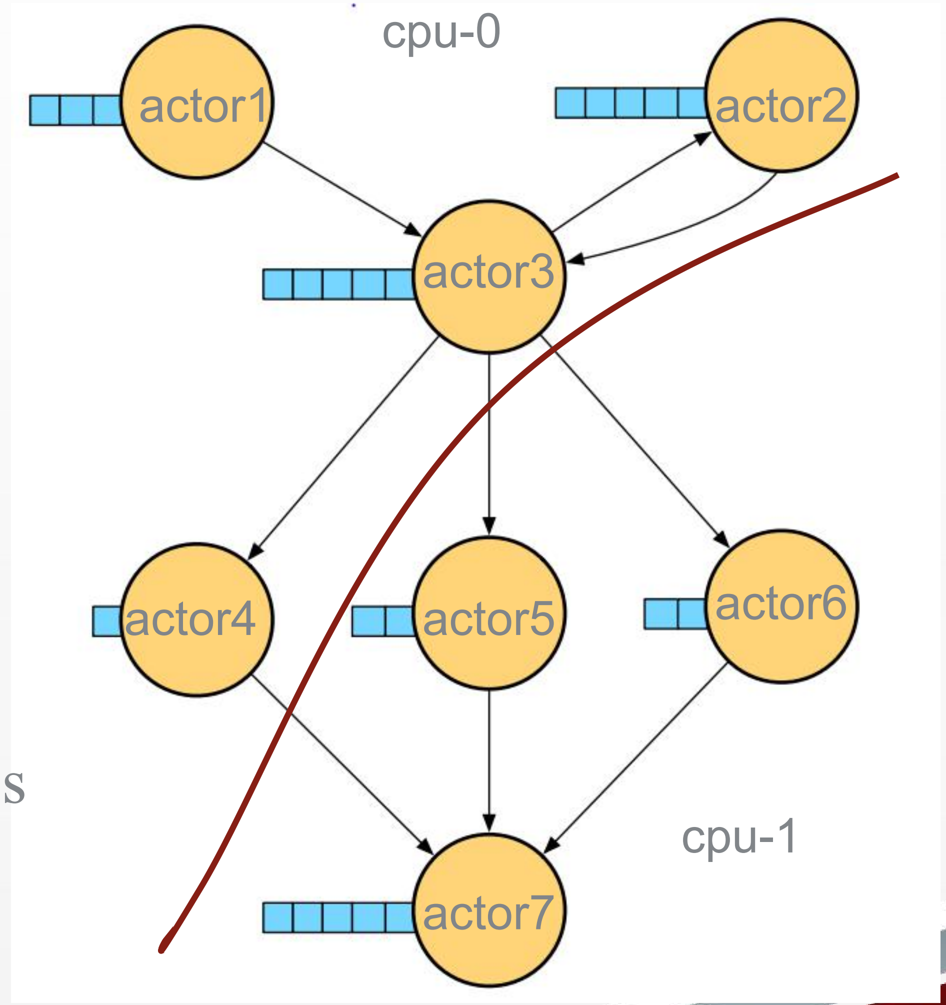
1): Actors and its connectivity form graph G=(V,E)

2): Actors can be partitioned to different processors

3): Actors can be any small tasks, for example,
neurals in neural network,
numerical integration task in FEM,
or particle push task in Monte Carlo calculation,
and many more.

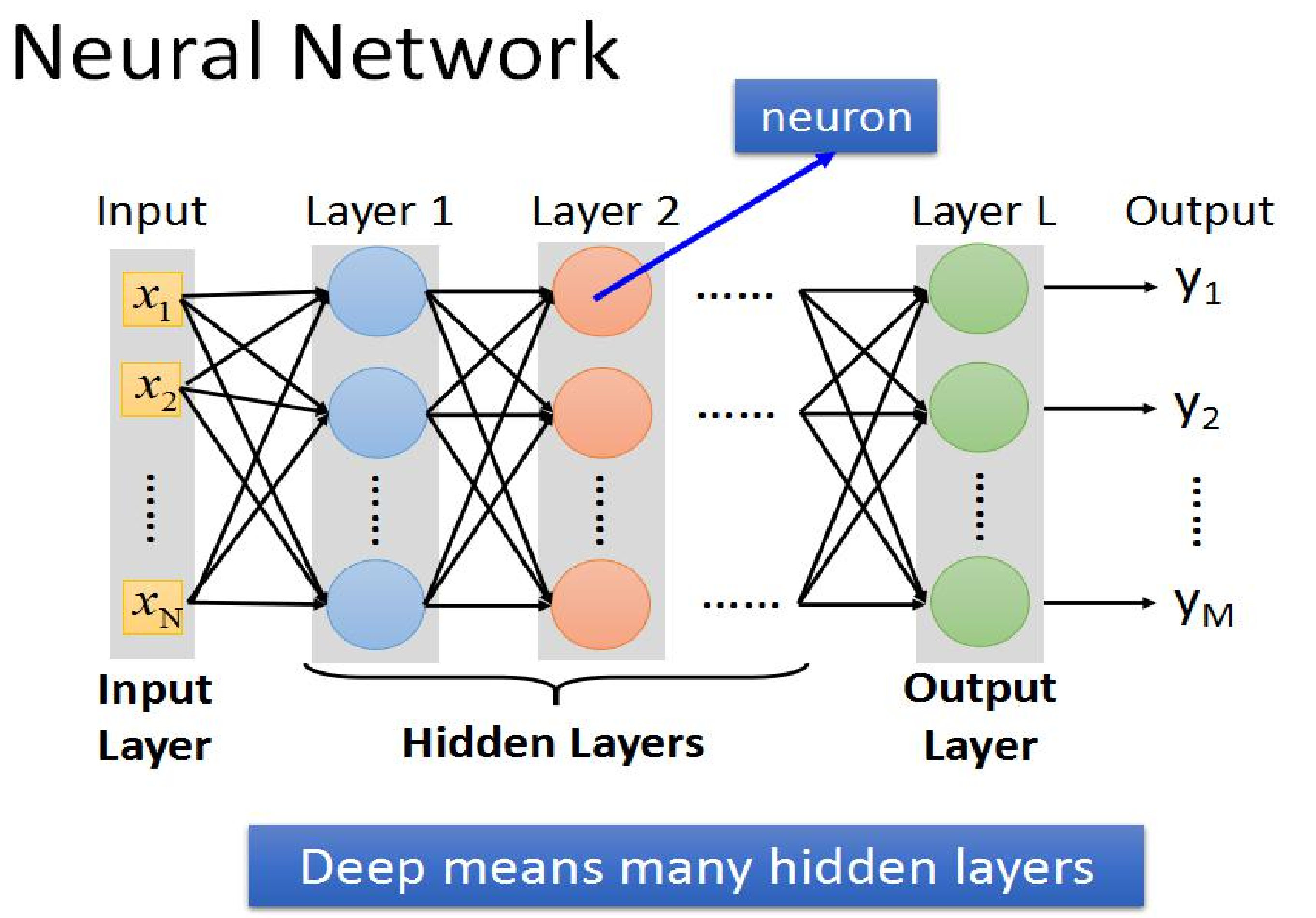
4): Actors can dynamically create/destroy daughtor actoress

5): asynchronous (non-block) communication



actor based DNN algorithm

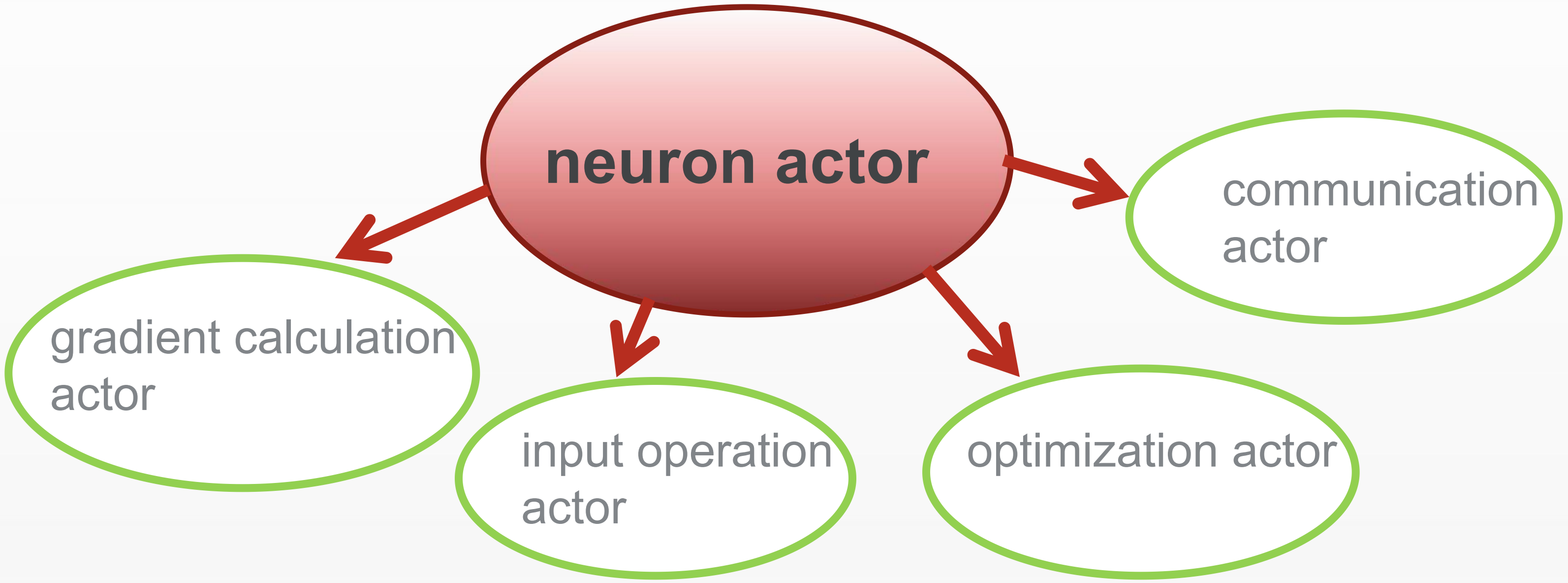
1): deep learning network is many hidden layers neural network
2): each layers have many foundmental unit that called neuron
3): neuron has input, output which form the connectivity
4): neurons and its connectivity form graph G=(V,E)



neuron is naturally an actor objects

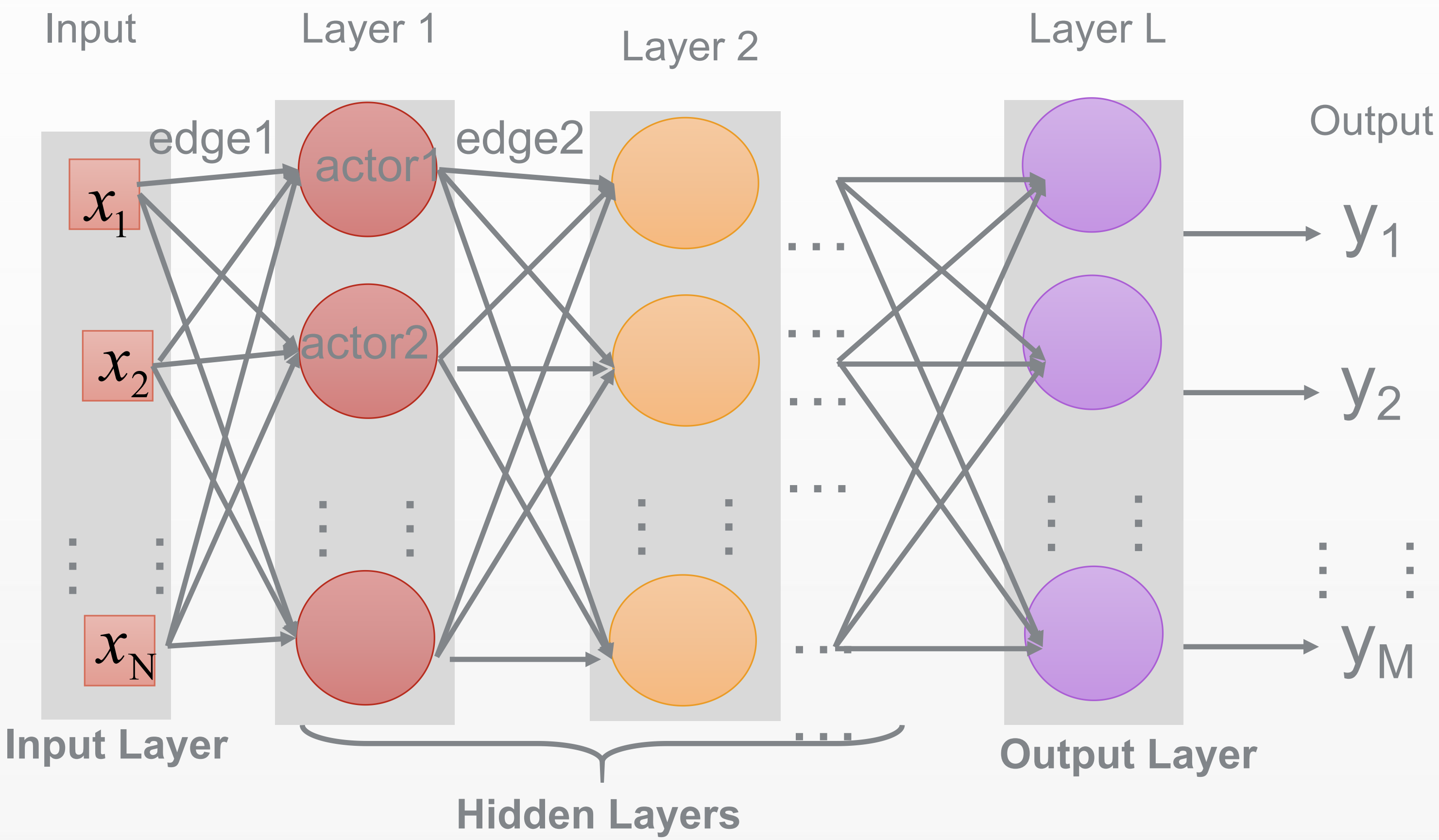
an actor model for neuron
(actor is the basic and foundmental class)

actor name
actor rank
processor ID
number of daughter actress
pointer to daughter actress
action (instructions)
number of threads
number of cores
connectivity



Each neuron actor can execute instructions in parallel, this arrangement give use many levels of parallelism.
All the neuron actor in the network form a graph G=(V,E), and can be partitioned to different computer node with minimum communication.

Put all neurons together to form a graph



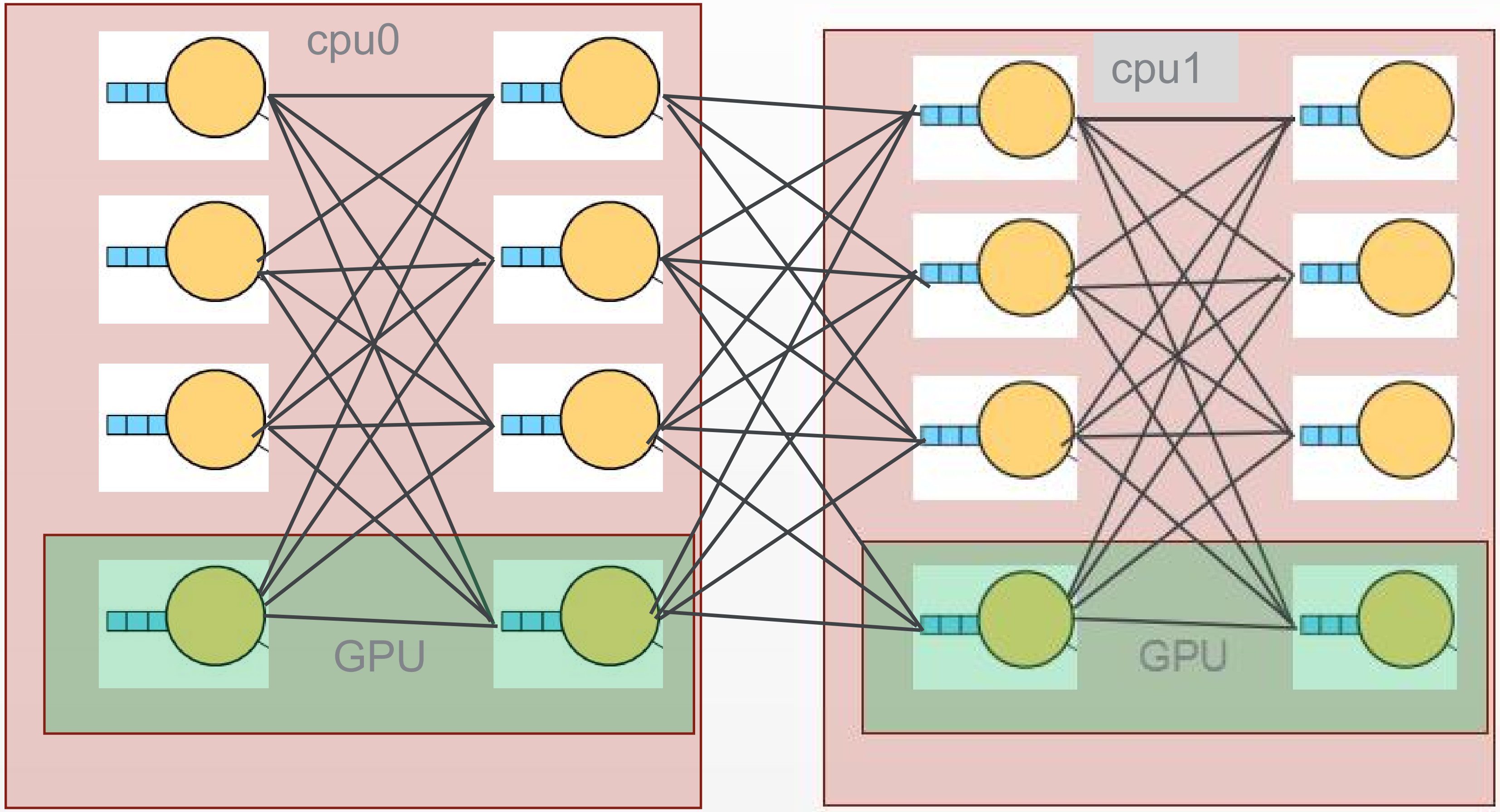
An actor is a node in graph,the connectivity is the edge (actor_1,actor_2,....actor_n,edge_1,edge_2,...edge_n)

Graph based partition for parallel computing

many ways to partition an unstructured mesh (metis,parmetis,scotch,pt-scotch)

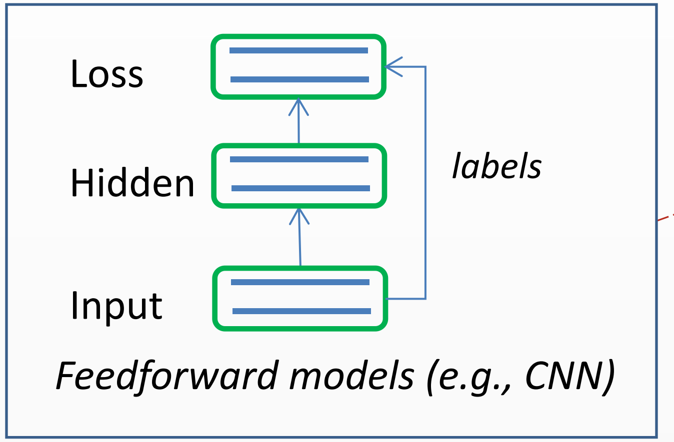
1): multilevel graph partitioning
2): recursive bisection graph partitioning
3): greedy algorithm
4): spectral partitioning
5): Kernighan-Lin algorithm

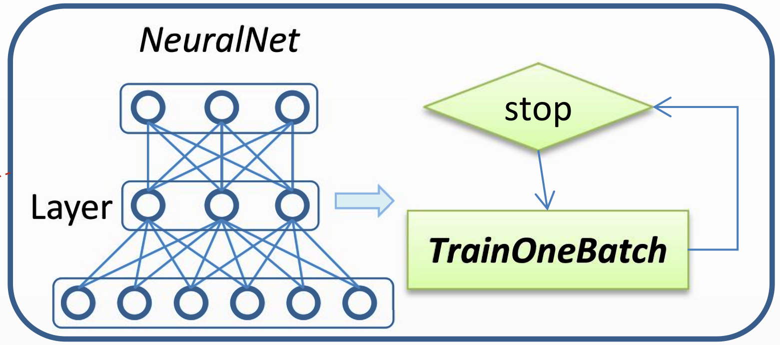
Put every actors together



1): Weighted actors can be used.
2): one actor can be mapped to two or more nodes if necessary.

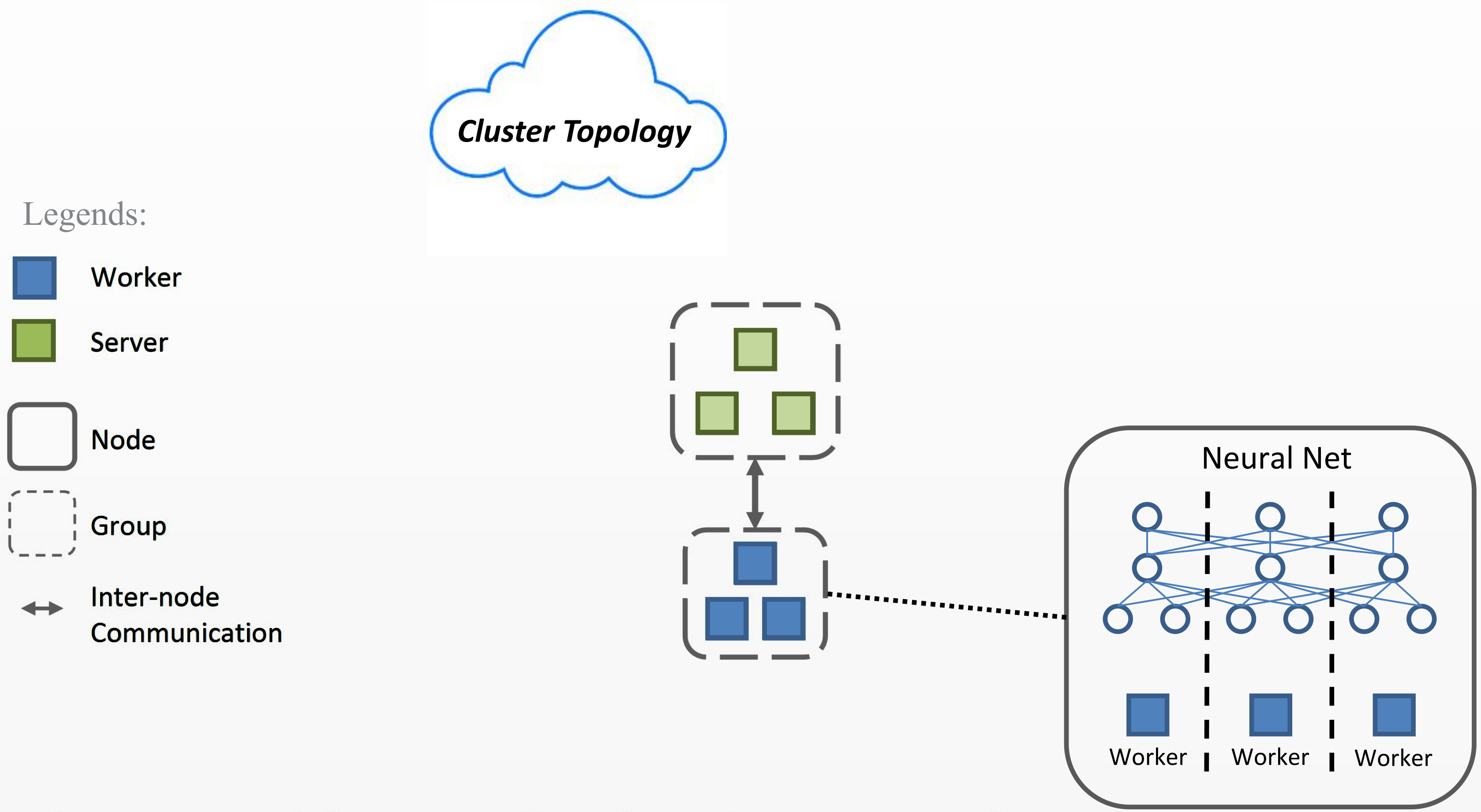
One-batch Training





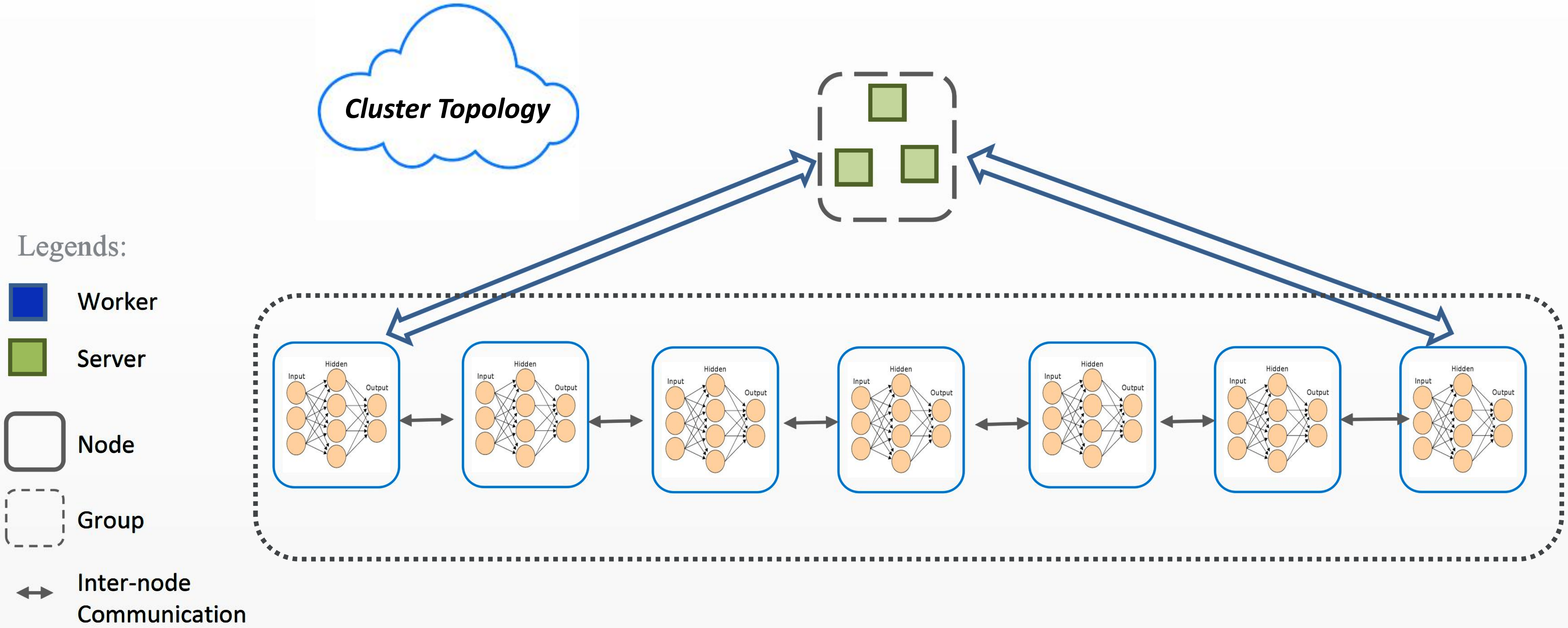
Back-propagation (BP)

Distributed online Training



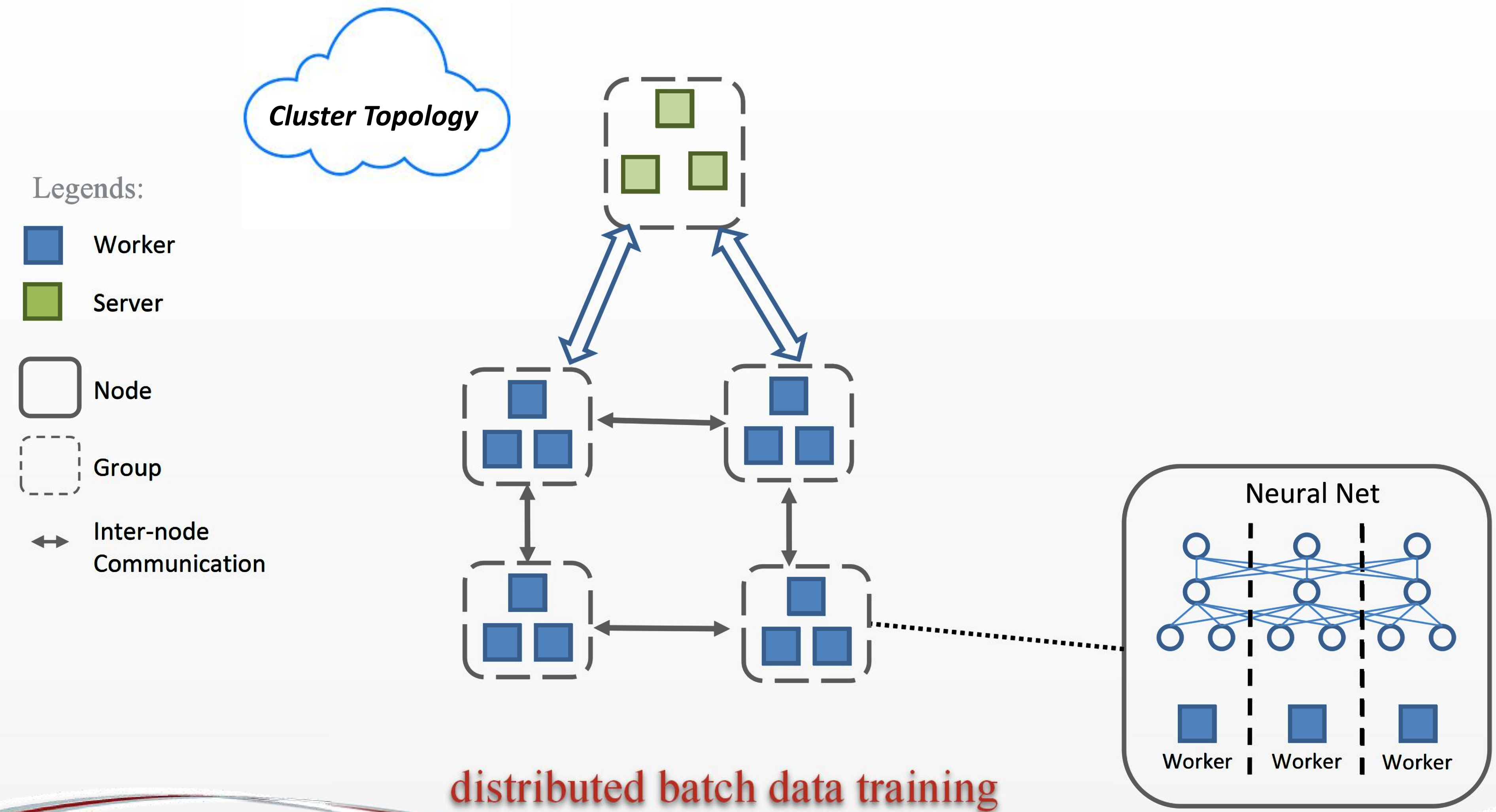
Synchronous training cannot scale to large group size

Distributed minibatch Training



each node trains one sample

Distributed minibatch Training



distributed batch data training

summaries and discussions

- An efficient way to implement deep neural network (DNN) can improve the network training.

- Using of deep neural network has the potential of speedup PT_SOLVER by a factor of 10-100, and better convergence an be expected without loss of accuracy

- Database need to be established based on the current PT_SOLVER TGLF runs.

- Many potential applications of DNN in fusion research, including DNN-EPED, experimental data analysis, and so on

the end

thank you very much !