Convolutional Deep Neural Networks on a GPU

by

Team Incognitos

Suhas **P**illai

Siddesh **P**illai

Recap

- Convolutional Neural Network (CNN) architecture.
- Sequential Approach.
- Parallel Approach.

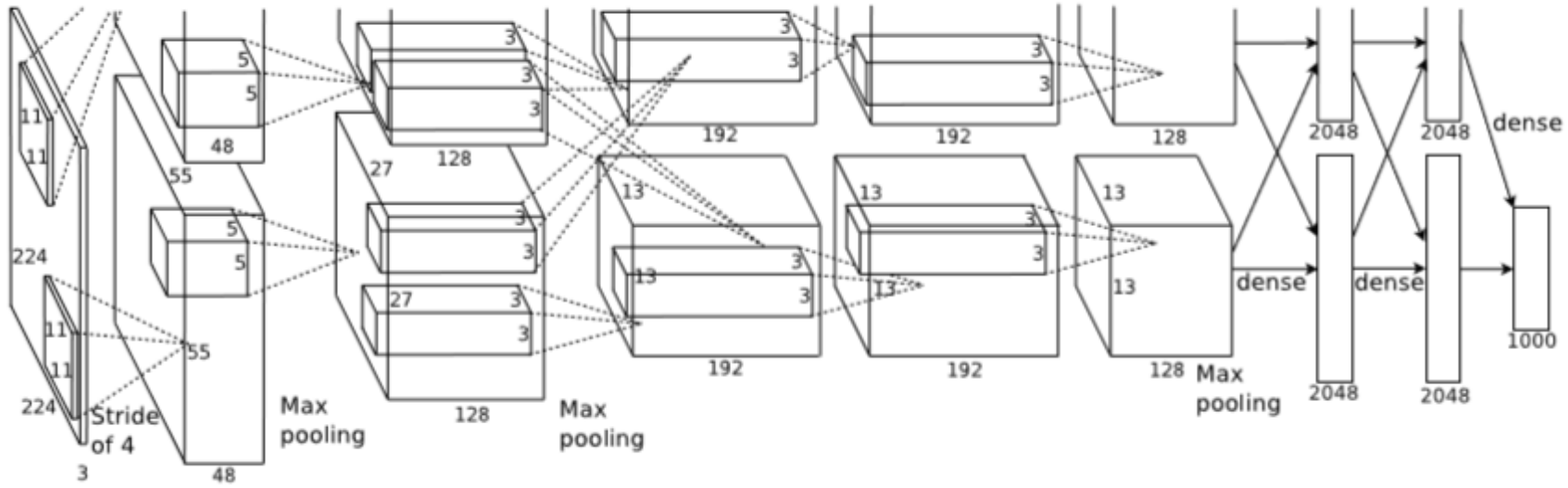
Research Paper - 1

- Authors : Alex Krizhevsky, IlyaSutskever, Geoffery E.Hinton.
- Title : ImageNet Classification with Deep Convolutional Neural Networks.
- Journal : Neural Information Processing Systems Conference (NIPS)
- Year : 2012
- Pages : 1-9
- URL : http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

Problems and new ways of training

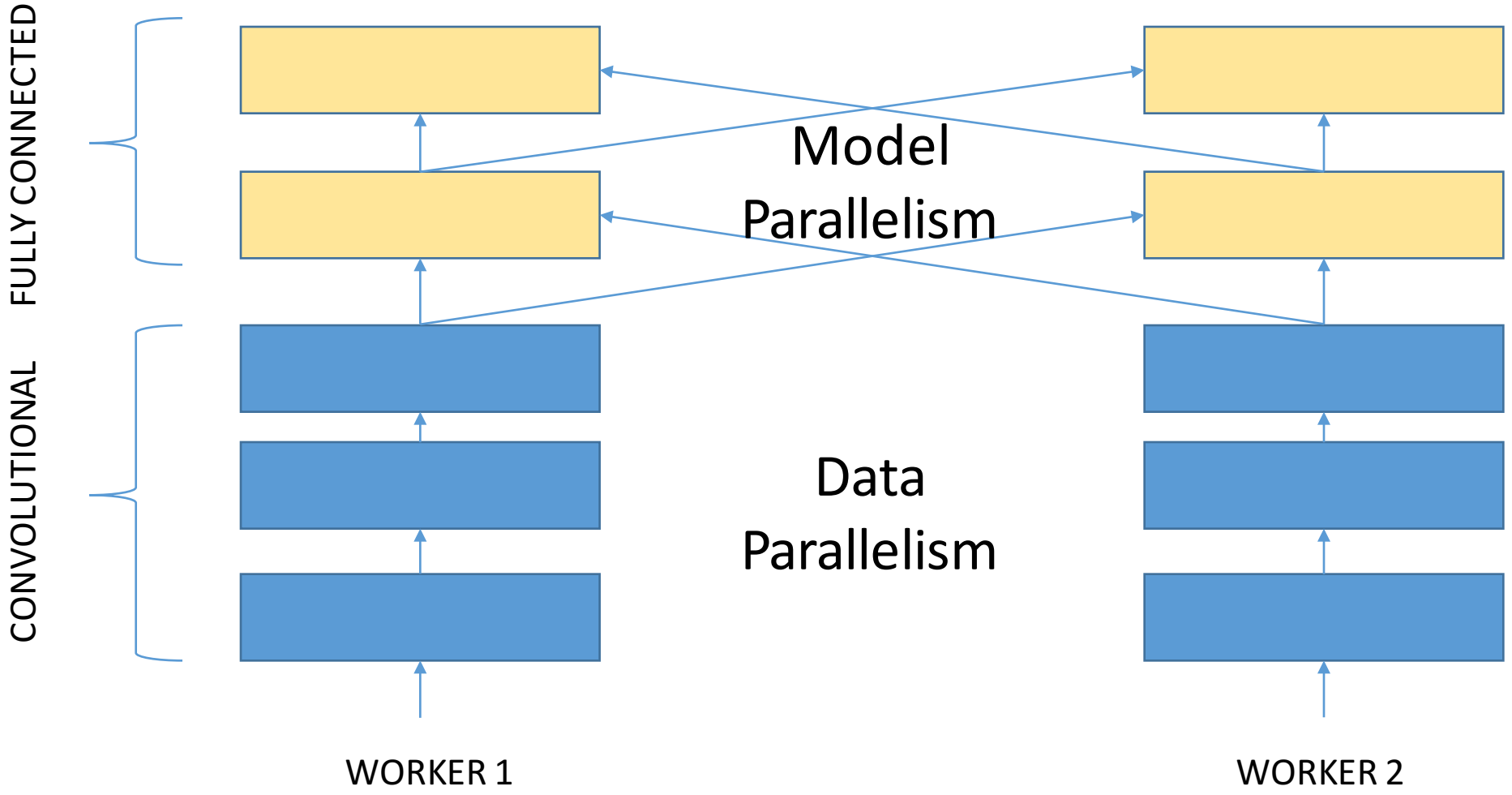
- Training deep convolutional neural networks requires lot of computing resources and lot of time.
- The paper suggests training deep networks on GPU with model parallelism.
- Model parallelism is done only in few selected layers.
- Solves problem of memory and also fast training of the network.

Alex Net Architecture



Source : ImageNet Classification with Deep Convolutional Neural Networks by Alex Krizhevsky.
http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

Architecture



Research Paper 2

- Authors : Adam Coates, Brody Huval, Tao Wang, David J. Wu, Andrew Y. Ng & Bryan Catanzaro.
- Title : Deep Learning with COTS HPC systems.
- Journal : Journal of Machine Learning Research.
- Year : 2013
- Pages : 1-9
- URL : http://jmlr.org/proceedings/papers/v28/coates13.pdf

Problems with Data and Model parallelism

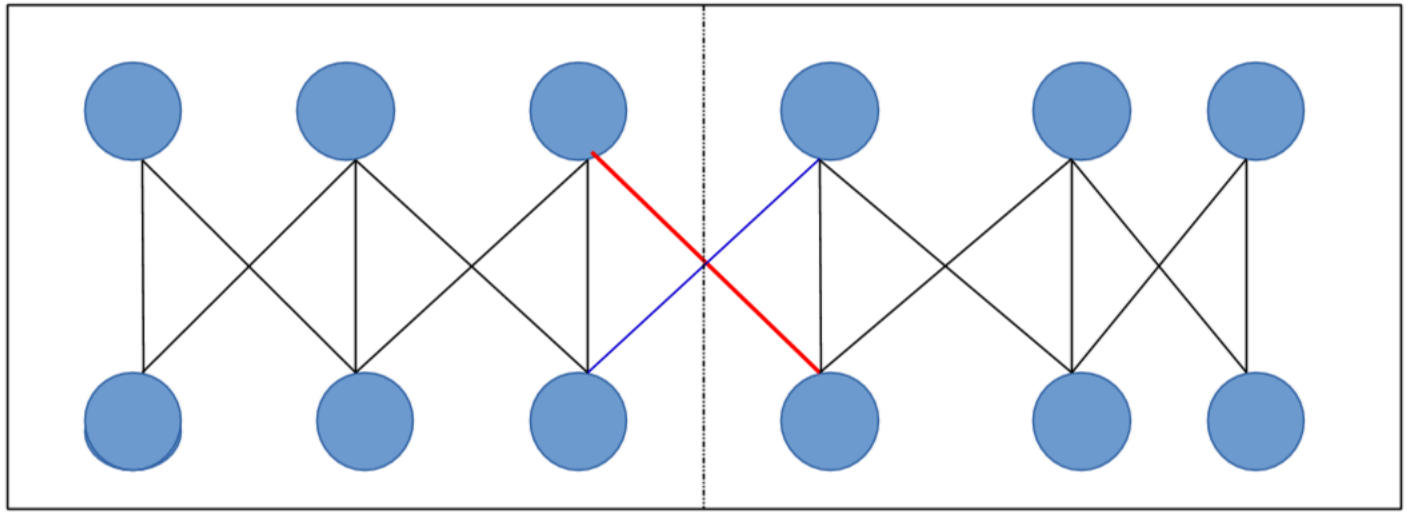
Data Parallelism :

- Synchronizing 1 billion parameters takes 30 sec.

Model Parallelism :

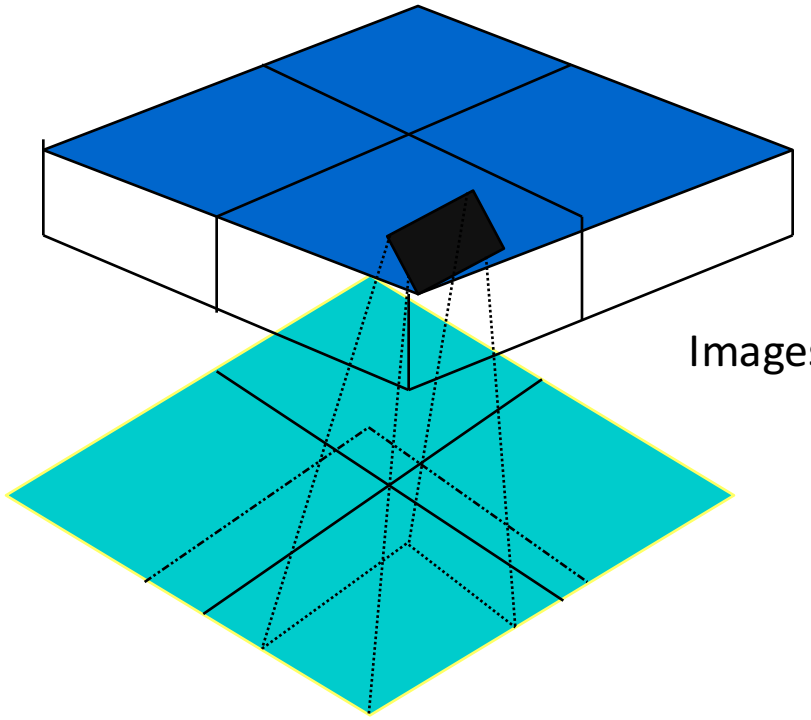
- More than 1 Mb of neurons for 100 images takes 0.8 sec.
- To tackle this latency, use InfiniBand for high performance computing with low latency and high throughput.
- MVAPICH2 : GPU aware MPI implementation, enables easy communications across machines.

Computations across machines



Machine 1 Machine 2

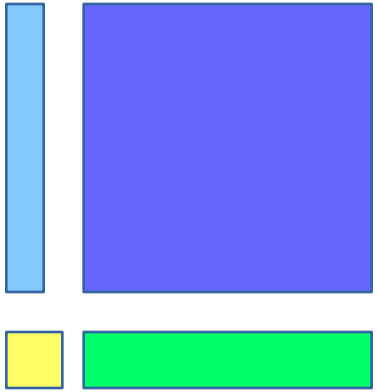
Distributed Array Abstraction



Images distributed across GPU's

Distributed Array Abstraction

- Neurons required for computations are stored in a buffered array on GPUs, which can be used for further computation



Research Paper 3

- Authors: Jeffrey Dean, Greg S. Corrado, Rajat Monga,Kai Chen, Matthieu Devin, Quoc V. Le, Mark Z. Mao,Marc'Aurelio Ranzato, Andrew Senior, Paul Tucker, KeYang, Andrew Y. Ng
- Title : Large scale distributed deep networks.
- Journal : Neural Information Processing Systems Conference (NIPS)
- Year : 2012.
- Pages : 1-9
- URL : http://papers.nips.cc/paper/4687-large-scale-distributed-deep-networks.pdf

Large Scale Distributed Deep Networks

- GPUs do not scale well when model does not fit GPU memory.
- To use GPU, researchers often reduce the size of data, so that CPU-GPU transfers are not significant bottlenecks.
- DistBelief software that supports model parallelism within machine and across machines.

Approaches for parallelism

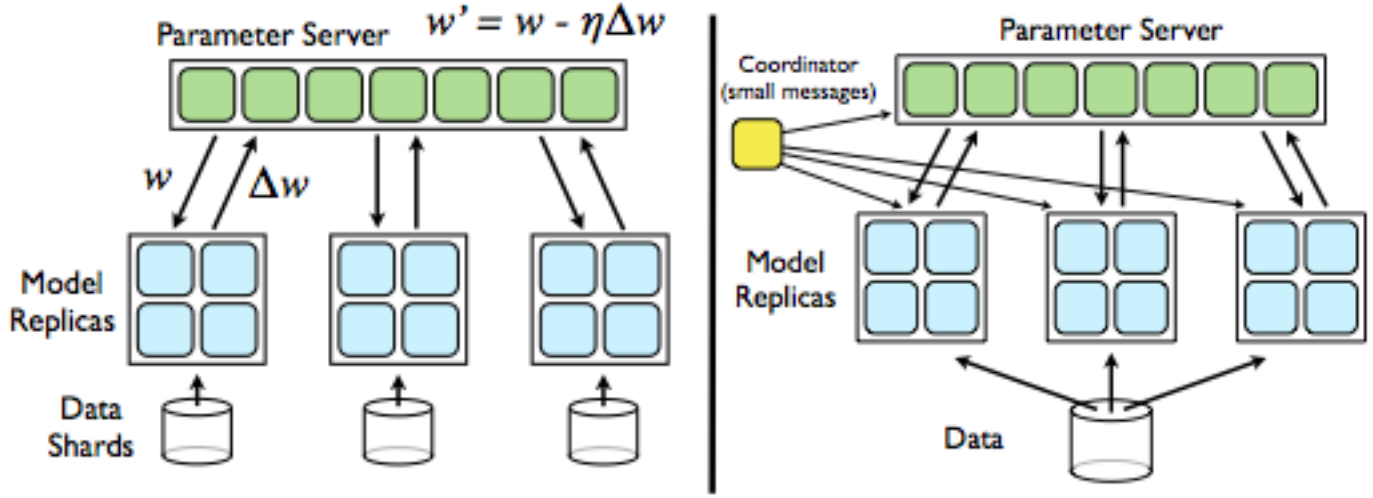
Two Algorithms

1. Downpour Stochastic Gradient Descent
2. Sand Blaster

Downpour SGD

- Centralized parameter server
- Data is divided into data shreds.
- Every model computes gradient for its data , then updates the centralized parameter server.
- Even if one model fails others can continue, hence asynchronous SGD.
- Some updates may be out dated while some may be very recent, but works remarkably well.

Downpour SGD and SandBlaster



Source : http://papers.nips.cc/paper/4687-large-scale-distributed-deep-networks.pdf

SandBlaster

- Data from batches divided and send across every machine
- Updates gradient after completion of one batch, hence less synchronization and less latency.
- There is a coordinator that issues command when any of the cores are idle.
- Fast and works equally well in comparison to Downpour SGD with Adagrad.

Thank you

Q.