SDA: Software-Defined Accelerator for Large-Scale DNN Systems

Jian Ouyang,[1] Shiding Lin,[1] Wei Qi, [1] Yong Wang, [1] Bo Yu, [1]

Song Jiang,[2]

[1]Baidu, Inc. [2]Wayne State University

Introduction of Baidu

- A dominant Internet company in China
  - ~US$80 Billion market value
  - 600M+ users
  - Exploiting internet markets of Brazil, Southeast Asia and Middle east Asia

- Main Services
  - PC search and mobile search
    - 70%+ market share in China
  - LBS(local base service)
    - 50%+ market share
  - On-line trips
    - QUNR[subsidiary company], US$3 billions market value
  - Video,
    - Top 1 mobile video in China
  - Personal cloud storage
    - 100M+ users, the largest in China
  - APPs store, image and speech

- Baidu is a technology-driven company
  - Tens of data centers, hundreds of thousands of servers
  - Over one thousand PetaByte data (LOG, UGC, Webpages, etc.)

DNN in Baidu

- DNN has been deployed to accelerate many critical services at Baidu
  - Speech recognition
    - Reduce 25%+ error ratio compared to the GMM (Gaussian Mixture Model) method
  - Image
    - Image search, OCR, face recognition
  - Ads
  - Web page search
  - LBS/NLP(Natural Language Processing)

- What is DNN (deep neural network or deep learning)
  - DNN is a multi-layer neural network.
  - DNN uses usually an unsupervised and unfeatured machine learning method.
    - Regression and classification
    - Pattern recognition, function fitting or more
  - Often better than shallow learning (SVM(Support Vector Machine), Logistics Regression, etc.)
    - Unlabeled features
    - Stronger representation ability
  - Often demands more compute power
    - Need to train much more parameters
    - Need to leverage big training data to achieve better results

Baidu 百度

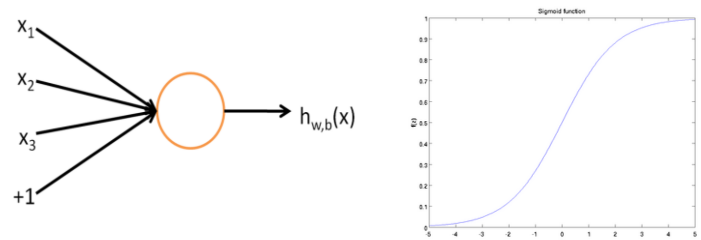
Outline

- Overview of the DNN algorithm and system
- Challenges on building large-scale DNN system
- Our solution: SDA (Software-Defined Accelerator)
  - Design goals
  - Design and implementation
  - Performance evaluation
- Conclusions

Bai du 百度

Overview of DNN algorithm

- Single neuron structure



$$h_{W,b}(x) = f(W^T x) = f(\sum_{i=1}^{3} W_i x_i + b) \qquad f(z) = \frac{1}{1+\exp(-z)}.$$

fig1

- Multiple neurons and layers

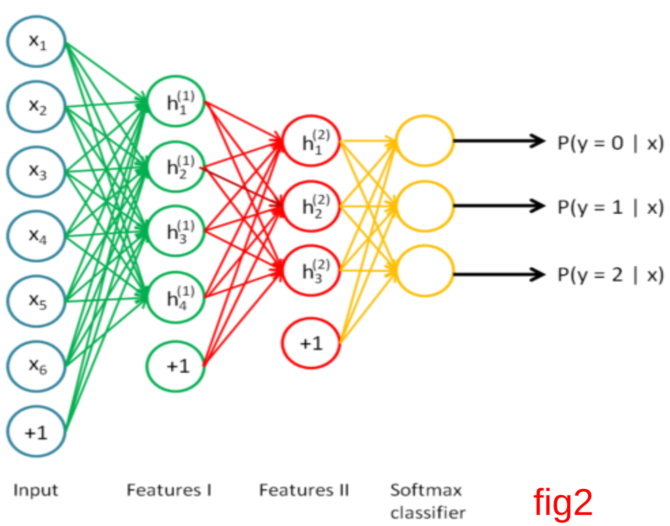


fig2

- Back-propagation training

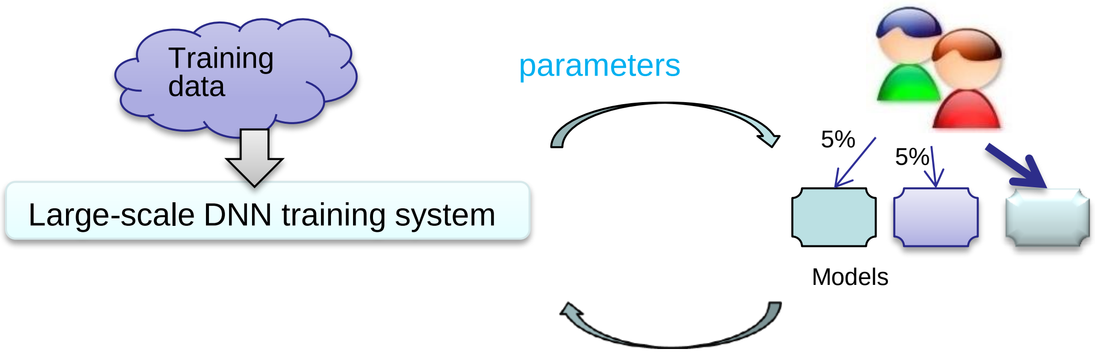
For each input vector
// forward , for input layer to output layer
O_i=f(W_i * O_i-1)
// backward, for output layer to input layer
delta_i = O_i+1 * (1-O_i) * (W_i * delta_i+1)
//update weight ,for input layer to output layer
W_i = W_i + n* delta_i*O_i-1
Almost matrix multiplications and additions
Complexity is $O(3*E*S*L*N^3)$
E: epoch number; S: size of data set; L: layers number; N: size of weight

- Online-prediction
  - Only forward stage

Online-prediction Complexity: $O(V*L*N^2)$
V : input vector number
L: layer number
N: size of weight matrix

N=2048,L=8,V=16 for typical applications, computation of each input vector is ~1GOP, and almost consumes 33ms in latest X86 CPU core.

Overview of DNN system



Off-line training

- Scale
  - 10~100TB training data
  - 10M~100B parameters
- workload type
  - Compute intensive
  - Communication intensive
  - Difficult to scale out
- Cluster type
  - Medium size (~100)
  - GPU and IB

On-line prediction

- Scale
  - 10M~B users
  - 100M~10B requests/day
- Workload type
  - Compute intensive
  - Less communication
  - Easy to scale out
- Cluster type
  - Large scale(K~10K)
  - CPU (AVX/SSE) and 10GE

Baidu 百度

Challenges on Existing Large-scale DNN system

- DNN training system
  - Scale: ~100 servers due to algorithm and hardware limitations
  - Speed: training time from days to months
  - Cost: many machines demanded by a large number of applications

- DNN prediction
  - Cost: 1K~10K servers for one service
  - Speed: latency of seconds for large models

- Cost and speed are critical for both training and prediction
  - GPU
    - High cost
    - High power and high space consumption
    - Higher demand on data center cooling, power supply, and space utilization
  - CPU
    - Medium cost and power consumption
    - Low speed

- Are any other solutions?

Bai du 百度

Challenges of large DNN system

- Other solutions
  - ASIC
    - High NRE
    - Long design period, not suitable for fast iteration in Internet companies
  - FPGA
    - Low power
      - Less than 40W
    - Low cost
      - Hundreds of dollars
    - Hardware reconfigurable
- Is FPGA suitable for DNN system ?

Bai du 百度

Challenges of large DNN system

- FPGA's challenges
  - Developing time
    - Internet applications need very fast iteration
  - Floating point ALU
    - Training and some predictions require floating point
  - Memory bandwidth
    - Lower than GPU and CPU

- Our Approach
  - SDA: Software-Defined Accelerator

SDA Design Goals

- Supports major workloads
  - Floating point: training and prediction
- Acceptable performance
  - 400Gflops, higher than 16core x86 server
- Low cost
  - Medium-end FPGA
- Not require changes of existent data center environments
  - Low power: less than 30w of total power
  - Half-height, half-length, and one slot thickness
- Support fast iteration
  - Software-Defined

Designs and implementations

- Hardware board design
- Architecture
- Hardware and software interface

Design – Hardware Board

- Specifications
  - Xilinx K7 480t
  - 2 DDR3 channels, 4GB
  - PCIE 2.0x8

- Size
  - Half-height, half-length and one slot thickness
  - Can be plugged into any types of 2U and 1U servers.
- Power
  - Supplied by the PCIE slot
  - Peak power of board less than 30w

Design - Architecture

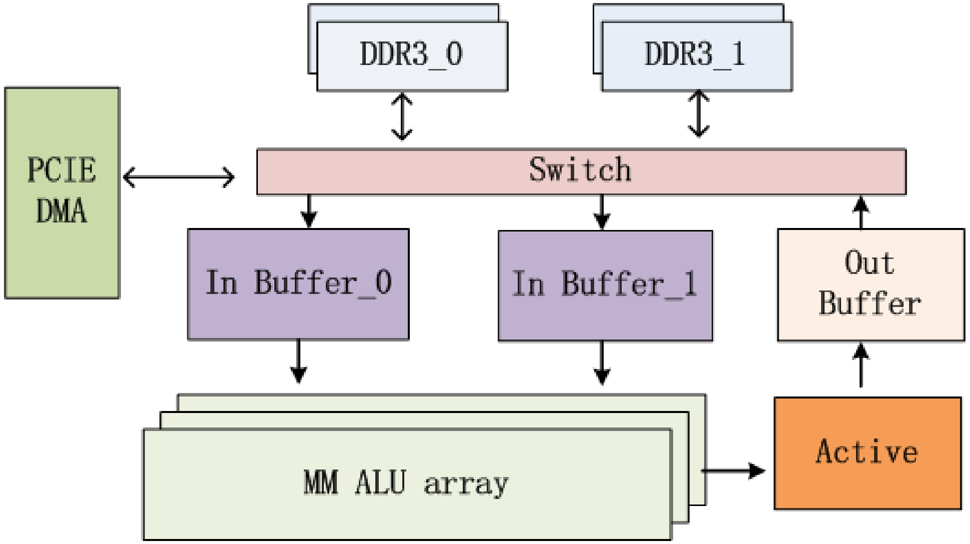
- Major functions
  - Floating point matrix multiplication
  - Floating point active functions

- Challenges of matrix multiplications
  - The numbers of floating point MUL and ADD
  - Data locality
  - Scalability for FPGAs of different sizes

- Challenges of active functions
  - Tens of different active functions
  - Reconfigurable on-line within milliseconds

Design - Architecture

- Customized FP MUL and ADD
  - About 50% resource reduction compared to standard IPs
- Leverage BRAM for data locality
  - Buffer 2x512x512 tile of matrix
- Scalable ALU
  - Each for a 32x32 tile



Design - architecture

- Software-defined active functions
  - Support tens of active functions: sigmod, tanh, softsign…
  - Implemented by lookup table and linear fitting
  - Reconfigure the table by user-space API

- Evaluations
  - 1-e5 ~1-e6 precision
  - Can be reconfigured within 10us

Design - Software/hardware Interface

- Computation APIs
  - Similar to CUBLAS
  - Memory copy: host to device and device to host
  - Matrix MUL
  - Matrix MUL with active function

- Reconfiguration API
  - Reconfigure active functions

Baidu 百度

Evaluations

- Setup
  - HOST
    - Intel E5620v2x2, 2.4GHz, 16 cores
    - 128GB memory
    - 2.6.32 Linux Kernel, MKL 11.0
  - SDA
    - Xilinx K7-480t
    - 2x2GB DDR3 on-board memory, with ECC, 72bit, 1066MHz
    - PCIE 2.0x8
  - GPU
    - One type server-class GPU
    - Two independent devices. The following evaluation leverages one device.

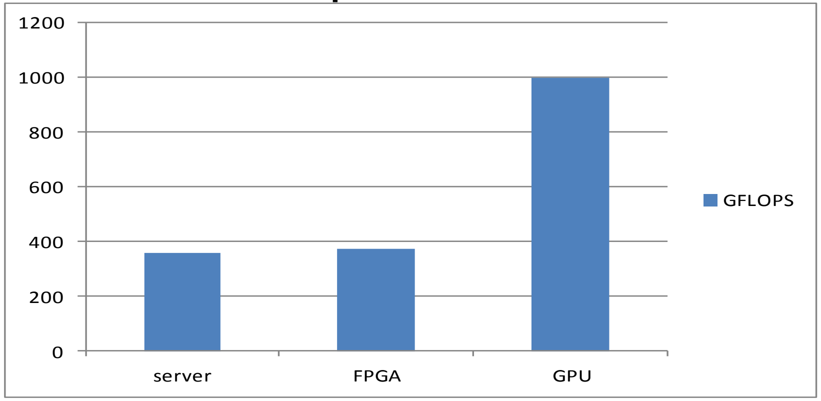
Bai du 百度

Evaluations-Micro Benchmark

- SDA implementation
  - 300MHz, 640 ADDs and 640 MULs

	LUT	DSP	REG	BRAM
Resource utilization	**70%**	**100%**	**37%**	**75%**

- Peak performance
  - Matrix multiplication : MxNxK=2048x2048x2048



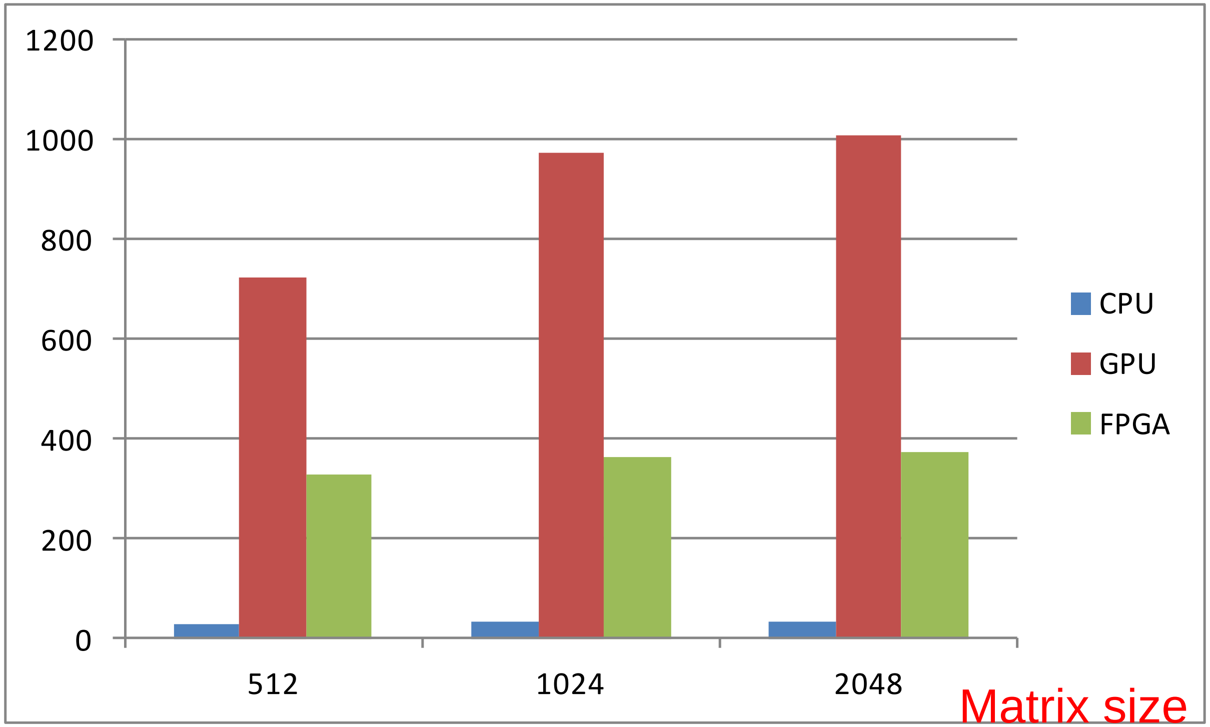
- power

	CPU	FPGA	GPU
Gflops/W	4	12.6	8.5

Evaluations-Micro Benchmark

- M=N=K, matrix multiplication
  - CPU leverages one core, GPU is one device
  - M=512,1024 and 2048

Gfops



Matrix size

Evaluations: On-line Prediction Workload

- Input batch size is small
  - Batch size: the number of input vector
  - Typical batch size is 8 or 16
- Typical layer is 8
- The size of hidden layer is several hundreds to several thousands
  - Depending on applications, practical tuning and training time
- Workload1
  - Batch size=8, layer=8, hidden layer size=512
  - Thread number is 1~64, test the request/s
- Workload2
  - Batch size=8, layer=8, hidden layer size=2048
  - Thread number is 1~32, test the request/s

Bai du 百度

Evaluations: On-line Prediction Workload

- Batch size=8, layer=8

- Workload1
  - Weight matrix size=512
  - FPGA is 4.1x than GPU
  - FPGA is 3x than CPU

Workload2
  - Weight matrix size=2048
  - FPGA is 2.5x than GPU
  - FPGA is 3.5x than CPU

- Conclusions
  - FPGA can merge the small requests to improve performance
  - Throughput in Req/s of FPGA scales better

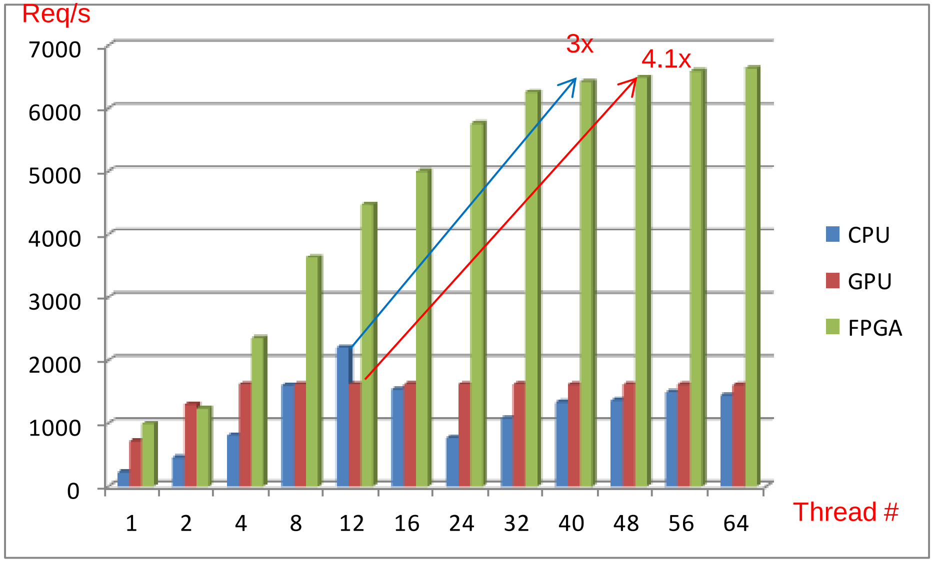


Fig a:workload1

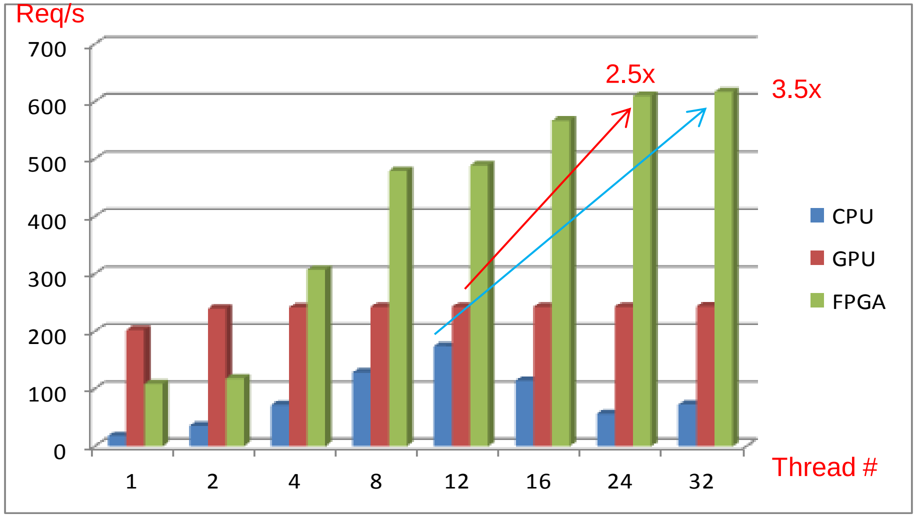


Fig b: workload2

The features of SDA

- Software-defined
  - Reconfigure active functions by user-space API
  - Support very fast iteration of internet services
- Combine small requests to big one
  - Improve the QPS while batch size is small
  - The batch size of real workload is small
- CUBLAS-compatible APIs
  - Easy to use

Conclusions

- SDA: Software-Defined Accelerator
  - Reconfigure active functions by user-space APIs
  - Provide higher performance in the DNN prediction system than GPU and CPU server
  - Leverage mid-end FPGA to achieve about 380Gflops
  - 10~20w power in real production system
  - Can be deployed in any types of servers
  - Demonstrate that FPGA is a good choice for large-scale DNN systems