AI in network = **Net For AI**

(NOT “AI for Net”)

Jeffrey He, **Rachel Chen**, Qiaoling Wang

July 19th, Montreal
One typical use case: DNN training platform

- **Distribution phase**: model parameters are distributed from the PS to all the workers
- **Aggregation phase**: workers calculate the gradients and send them to the PS (parameter sever)
- The traffic models of the two phases are one-to-many and many-to-one
- Reducing network traffic between PS and switch can accelerate the training
- Accelerating DNN training is very important, since one training takes up to tens of days

<table>
<thead>
<tr>
<th>CNN</th>
<th>Parameter Quantity (MB)</th>
<th>Single Iteration Time (s)</th>
<th>Total Training Time (Day)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AlexNet-v2</td>
<td>192</td>
<td>13.0</td>
<td>18.8</td>
</tr>
<tr>
<td>Inception-v3</td>
<td>106</td>
<td>4.3</td>
<td>6.23</td>
</tr>
<tr>
<td>Resnet-152</td>
<td>230</td>
<td>12.3</td>
<td>17.8</td>
</tr>
<tr>
<td>VGG-16</td>
<td>528</td>
<td>29.0</td>
<td>42.0</td>
</tr>
</tbody>
</table>

Note: 1 PS + 32 workers, batch size=32, 1 epoch = 125114 iters, 100 epochs in total

Computing and multicast reduce network traffic

- Data transmission between PS and workers takes the majority of the time.
  - Take a 10GE-link as an example, the transmission time is up to dozens of times longer than the calculation time in one single iteration. (eg, VGG-16)

<table>
<thead>
<tr>
<th>DNN model</th>
<th>Parameter Quantity (MB)</th>
<th>Calculation time (ms)</th>
<th>Single iteration (ms)</th>
<th>Theoretical Transmission (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inception-v3</td>
<td>106</td>
<td>1035.7</td>
<td>1700</td>
<td>10Gbps 1017.6 254.4</td>
</tr>
<tr>
<td>Resnet-152</td>
<td>230</td>
<td>650.9</td>
<td>2781</td>
<td>2208 552</td>
</tr>
<tr>
<td>VGG-16</td>
<td>528</td>
<td>285.2</td>
<td>7114</td>
<td>5068.8 1267.2</td>
</tr>
</tbody>
</table>

- Scott Shenker’s team (from UC Berkeley) add in-network computing and multicast to reduce the E2E training time of VGG-16 to 1/29 (when 32 servers were deployed) [1].

<table>
<thead>
<tr>
<th>DNN Name</th>
<th>Baseline</th>
<th>Mult/Agg</th>
</tr>
</thead>
<tbody>
<tr>
<td>VGG-16</td>
<td>42.08 sec</td>
<td>1.43 sec</td>
</tr>
<tr>
<td>Inception-v3</td>
<td>4.77 sec</td>
<td>1.05 sec</td>
</tr>
<tr>
<td>Resnet-200</td>
<td>13.89 sec</td>
<td>0.96 sec</td>
</tr>
<tr>
<td>Resnet-101</td>
<td>9.09 sec</td>
<td>1.01 sec</td>
</tr>
</tbody>
</table>


Theoretically, when introduce in-network computing and multicast (involves 32 workers), we can still use cheap 10GE switch rather than expensive 100GE.
Why the E2E training time greatly reduced?

Traffic flow:
1) from workers to switch
2) from switch to PS

Traffic from workers to switch:
- Wn upload Traffic: $\Delta = 0.1$
- ...: $\Delta = 0.2$
- W2 upload traffic: $\Delta = 0.3$
- W1 Upload Traffic: $\Delta = 0.4$

Traffic from switch to PS:
- with “perfect” network scheduling and shaping
- $\Delta = 0.1 \ 0.2 \ 0.3 \ ... \ 0.4$

Traditional way:
switch as traffic mover

Add “computation”: in-net aggregation and transfer results to PS, reducing traffic from switch to PS.

Add “cache”: store model parameters and distribute to workers, reducing network traffic from PS to switch.
Recent advances in research: Computing

- Amedeo Sapio, Marco Canini, etc. "In net computing is a dumb idea whose time has come ", Hotnets 2017
  - A large data reduction ratio (86.9%-89.3%) and a similar decrease in workers’ computation time
  - [https://dl.acm.org/citation.cfm?id=3152461](https://dl.acm.org/citation.cfm?id=3152461)

- Michael Alan Chang, Scott Shenker, etc. “ChangNetwork Evolution for DNNs”, SysML, Feb 2018, Palo Alto, California
  - Optimizing the network fabric can improve DNN training time
  - [https://www.sysml.cc/doc/182.pdf](https://www.sysml.cc/doc/182.pdf)

  - Improvement of a factor of 3.24 in the latency of a 4096 byte MPI_Allreduce() operations, declining from 46.93 to 14.48 microseconds.

- NetCompute 2018: In-Network Computing workshop in sigcomm2018. (this August)
Recent research: introduce other capabilities to network

• In network cache (maybe obtain the capability similar to the multicast):

  • Xin Jin etc, “NetCache: Balancing Key-Value Stores with Fast In-Network Caching”, SOSP2017,(UCB: Ion Stocia team)
    • The throughput increases by 3-10 times. The query delay of 40% can be shortened by 50%.
  • Xiaozhou Li etc," Be fast, cheap and in control with SwitchKV", (Princeton) NSDI'2016,
    • Increases the throughput by 5 times and improves the delay by 3 times.
    • https://dl.acm.org/citation.cfm?id=2930614

• In network consensus

  • Huynh Tu Dang etc, "Paxos Made Switch-y", sigcomm CCR 2016,(Marco Canini team)
    • www.sigcomm.org/sites/default/files/CCR/papers/2016/April/0000000-0000002.pdf
  • Zsolt István etc," consensus in a box", NSDI 2016,
    • https://dl.acm.org/citation.cfm?id=2930639
  • Dan R. K. Ports etc, "Designing Distributed Systems Using Approximate Synchrony in Data Center Networks", (University of Washington)NSDI 2015.
  • Jialin Li etc, “Eris: Coordination-Free Consistent Transactions Using In-Network Concurrency Control”,(University of Washington) SOSP 2017.
Looks Promising? But...

- Adding “X” functions into network violates “end-to-end argument”
  - be cautious to add new functions if they are not used by majority of applications
  - Impact the TCP transport, security and privacy...

- Or, maybe we can
  - avoid to develop ad hoc solutions
  - develop new abstraction of data plane and new architecture to make full use of new programmable networks
  - Meanwhile, these capabilities are generic enough and decoupled from specific applications
  - Trial it firstly in constrained area of networks like DCN
Also keep in mind...

• Existing programmable network devices were designed for forwarding, so limited for "X" functions
  • the size of cache table
  • the number of bytes that the devices can manipulate on each packet
  • floating points
  • etc...

• Probably a good research topic
  • jump out from the off-shelf devices
  • find out the “sweet spot” between the pain(extra cost) and the gain
  • provide requirements for next generation of programmable devices
  • etc ...
Any comments are welcomed.

For further discussion/ comments

chenlijuan5@Huawei.com