Amber: Large-Scale Deep Learning for Intelligent Computer System
Use of Machine Learning at WeChat

# of directories of containing model files

Directories

Time


0  50  100  150  200  250

products/areas

text classification
sentiment analysis
topic analysis
conversation generation
Event detection
spam recognition
User modeling
robotics
image understanding
face recognition
online advertising
speech recognition
wechat search
OCR
augmented reality
security defense

...many others...
Outline

Two generations of machine learning software systems

- 1st generation: Amber-Trex
- 2nd generation: Amber-Velo
Varied raw data

- **Text data:** trillions of words of Chinese + other languages
- **Visual data:** billions of images and videos
- **Audio:** tens of thousands of hours of speech per day
- **User activity:** post, comment, thumbs up, emotion icon, queries, collection, transmit, ignore etc.

...
Applications @ image
Applications @ audio
Applications @ NLP
Applications @ User profiling
Applications @ User profiling
Applications @ User profiling
Stay true to the mission

To explore what is possible or better in perception, cognition and language understanding

New
Efficient
Algorithm
Computation
Unique
Architecture
Reasonable
Insights
Active
Imagination
Large
Datasets
Open source projects about Deep Learning

- **Caffe**: the first mainstream industrial grade DL tool;
- **Purine**: composite graph;
- **TensorFlow**: separation of op and its gradient, flexible, reproducibility, portability;
- **MXNet**: programming paradigm fusion, w/r dependent concurrency, memory reuse in place, memory analysis in stages, subgraph optimization;
- **CNTK**: automatic loop unwrapping, NDL, MEL, 1-bit SGD, adaptive minibatch;
- **Theano/Torch**: many research ideas come from them, flexible graph/layer description, github launch earlier;
- ...
Train Large, Powerful Models Quickly

- Data/model parallelism
- Flexible resource management and scheduling
- Compile the graphs
- Best-effort concurrent operations
- Limited memory reuse
- Consistent data streaming
- Kernels merge
Amber-Velo Framework
Client

- Symbolic network declaration
- All parallel (data/model) logic is completed in a process
- Only need to focus on logic of the algorithm
- Asynchronous and synchronous interface
- Dynamic interaction with Master
- C++ & python front ends

```python
import amber as am

def alloc_vars():
    with am.Device("/ps:s0/device:CPU:0"):
        w1 = am.Variable("fc1_w", shape=(128, 784))
        w2 = am.Variable("fc2_w", shape=(10, 128))
        return [w1, w2]

def get_mlp(w1, w2, input):
    with am.Device("/worker:w0/device:GPU:0"):
        fc1 = am.symbol.FullyConnected(data=input[0], params=[w1])
        act1 = am.symbol.Activation(data=fc1, act_type="relu")
    with am.Device("/worker:w1/device:GPU:1"):
        fc2 = am.symbol.FullyConnected(data=act1, params=[w2])
        mlp = am.symbol.SoftmaxOutput(data=[fc2, input[1]])
    return mlp

def train_net(input):
    [w1, w2] = alloc_vars()
    mlp = get_mlp(w1, w2, input)
    handler = am.GetMaster().Compile(mlp)
    for i in range(0, 1000):
        handler.run()
        if i%100 == 0:
            print handler.GetBlob(mlp.GetId(0), is_train=True)
```
Master

- Compiling optimization of graphs
  - Consanguinity in subgraphs
  - Server load balancing placement
  - Composite graph and its optimization
  - Memory initialization description and reuse analysis
  - ...

- Stale control, Monitoring status

- Special optimization for sparse structure
**Scheduler**

Yard

- Self-developed resource management & schedule
- Exception handling and recovery automatically
- Resource dynamically scaling
- Assignable processes exist in same machine preferentially
- Warm start and resume from breakpoint
- Start run dependencies between jobs,
  
  e.g.

  A \leftrightarrow B \rightarrow E
  
  C \leftrightarrow D
Worker

- Asynchronous data reader
  - Read ahead / read caching
  - multiple parsers & user-defined parser
  - Automatic partition and shuffle
  - variable-length structure and auto-completion
- Push & pull based on priorities
- Best-effort concurrency
- Decoupling computation and network
- Multiple Ops write the same tensor
Parameter server

- Decouple network and logic to facilitate integrate new technology
- Asynchronous abstraction of localPs interface
- Interactive data persistence
- Tradeoff between full steam and merging with same priority
- BSP/SSP/ASP and hybird
Debug & Monitoring

Facilitate innovation and experiment

- Examine symbolic description at any time
- Batch or Ops granularity execution progress
- Obtain the output of any Op or tensor interactively
- Trend dynamic display
  - Interactive training to comprehend hyper-parameters
  - To find the relative degree of a parameter variation
  - To analyze which parameters or ops are critical and which are redundant
Experiments

Run the experiments on the following configuration:

1. Tesla K40c, CUDA 7.0, CUDNN 4.0.0.7
2. FB Memory: 11519 MiB, Clocks Memory: 3004 MHz, Performance State: P0
3. Intel(R) Xeon(R) CPU E5-2650 v3 @ 2.30GHz, 125GB memory
4. 10G Ethernet
5. MXNet 0.7.0; Tensorflow 0.10
Experiments – AlexNet[1]

Experiments – GoogLeNet[2]

Experiments – VGG[3]

Experiments – LSTM[4]

Thank you