



# Pluto

## A Distributed Heterogeneous Deep Learning Framework

Jun Yang, Yan Chen  
Large Scale Learning, Alibaba Cloud

A decorative graphic element on the left side of the slide, consisting of a series of concentric, curved lines that form a funnel-like shape, pointing towards the right.

## Outline

- **PAI(Platform of Artificial Intelligence)**
  - PAI Overview
  - Deep Learning with PAI
  - Pluto
- PAI DL Application
  - Chatbot Engine
- Summary

# Machine Learning Platforms



Amazon Machine Learning

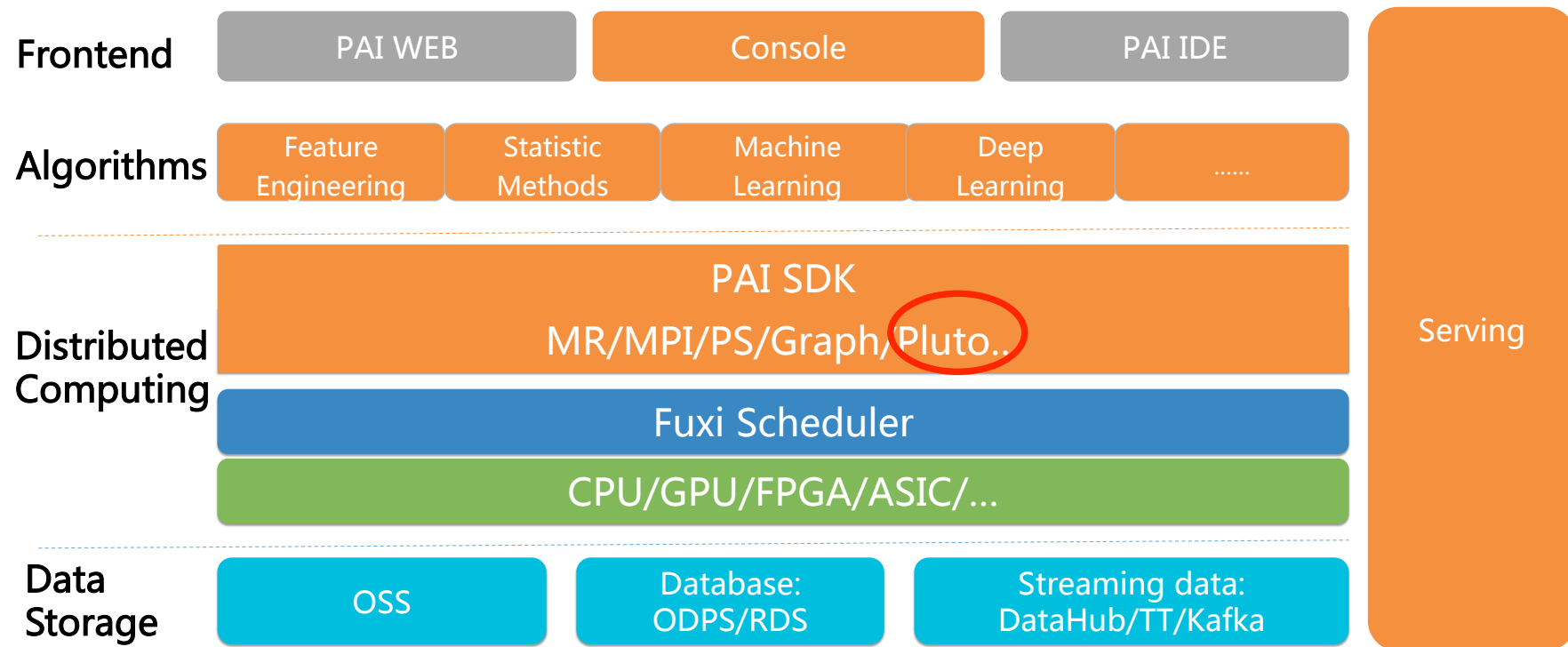


Microsoft Azure Machine Learning



Google Cloud Platform

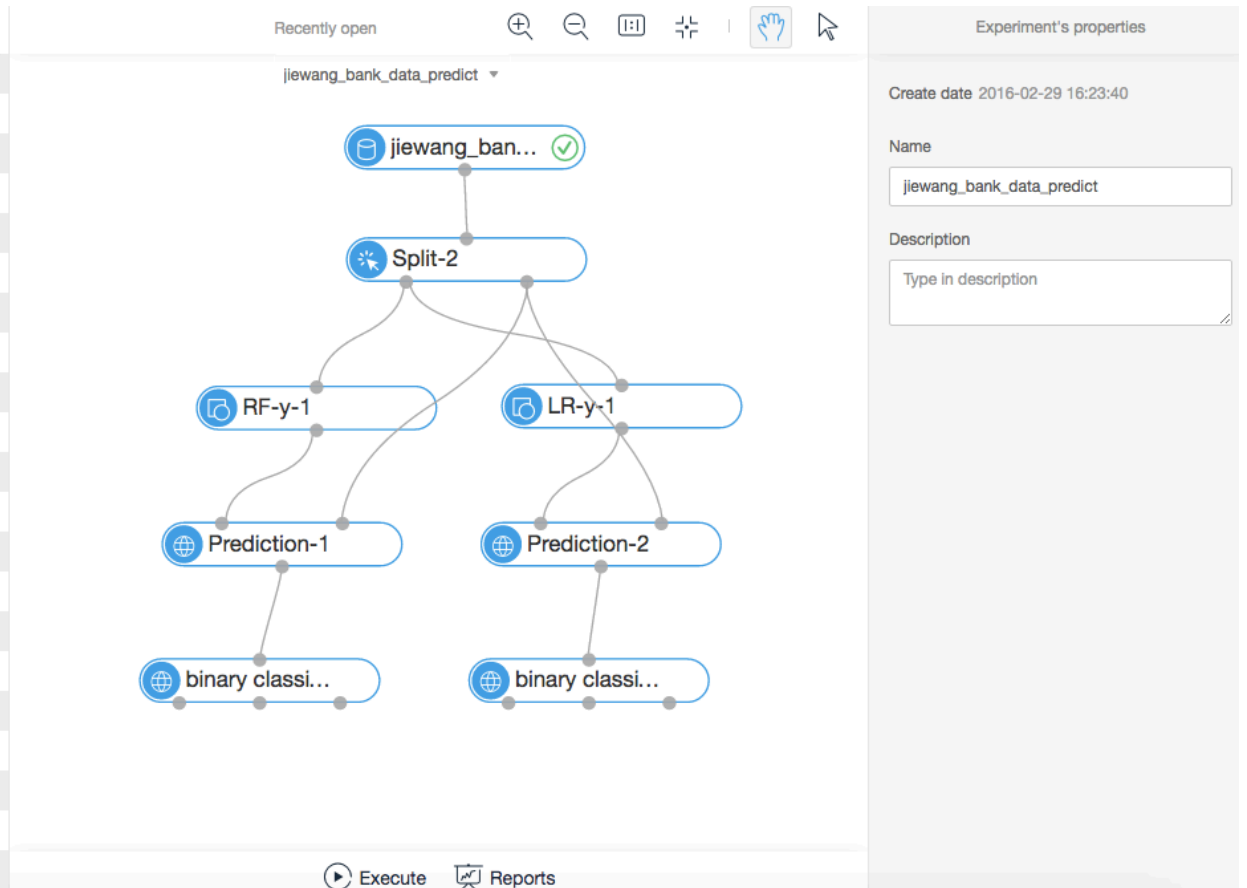
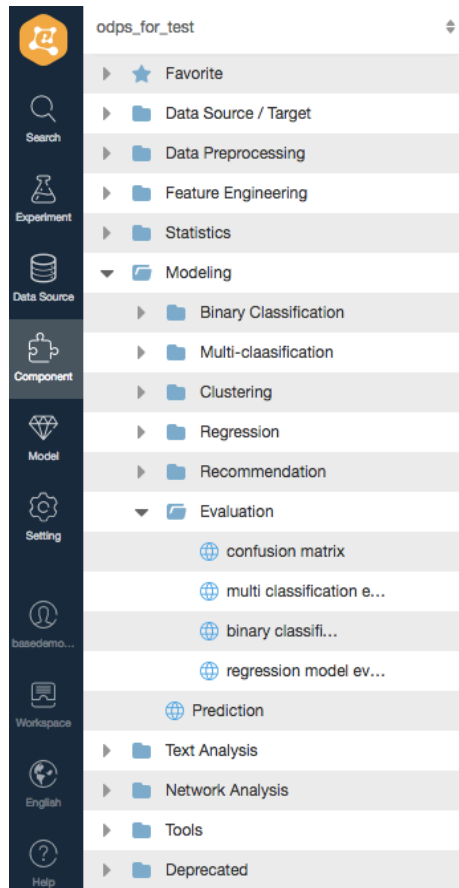
# PAI Overview



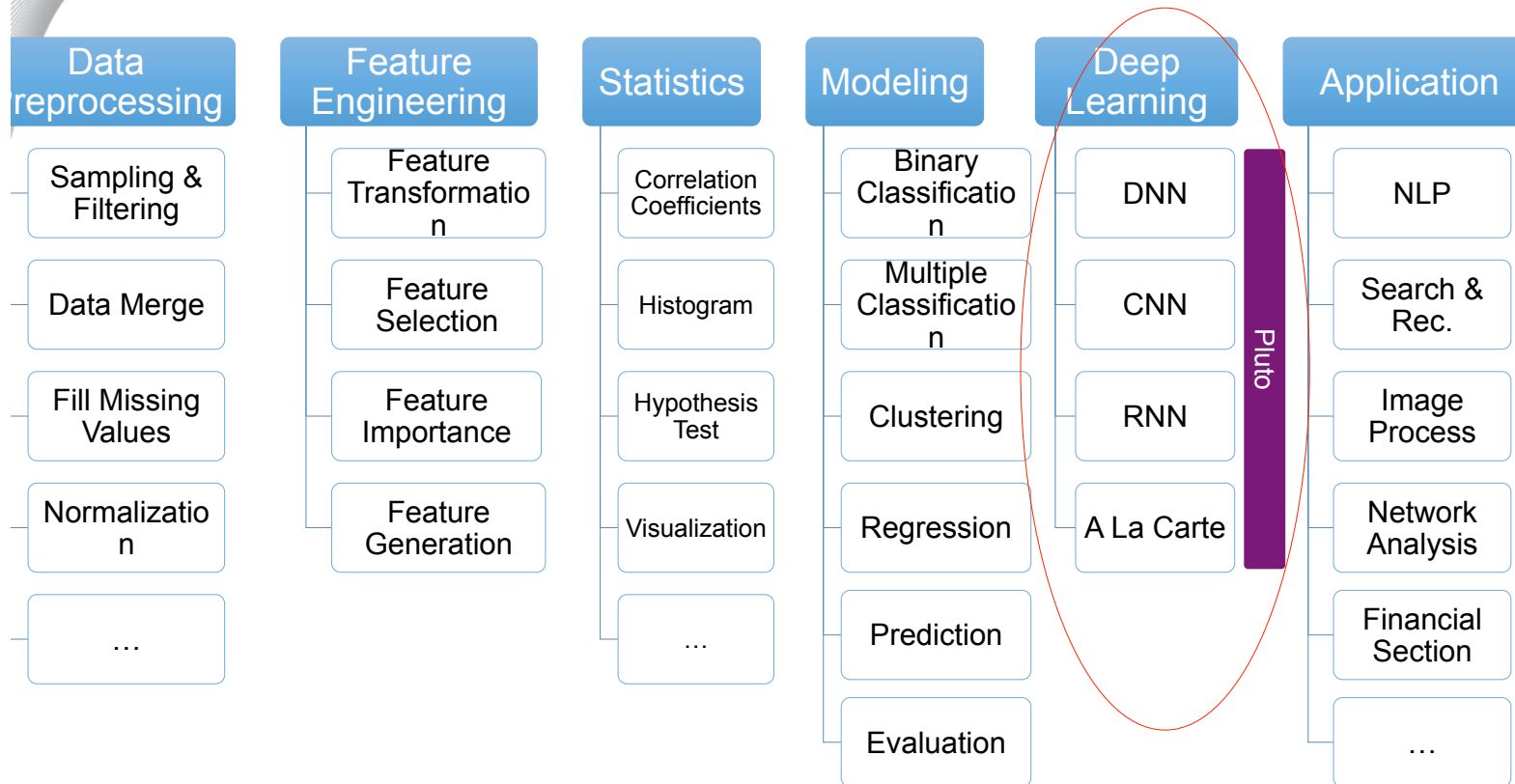
Tutorial: [data.aliyun.com](https://data.aliyun.com)

# PAI Project

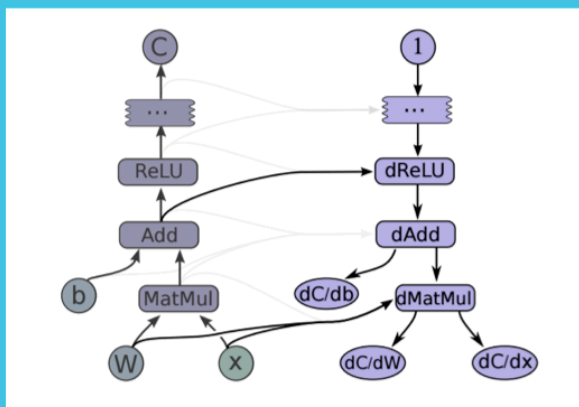
Search  
Experiment  
Data Source  
Component  
Model  
Serving



# Machine Learning with PAI



# Deep Learning with PAI



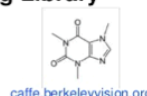
Caffe

Based on tutorials by the Caffe creators at UC Berkeley

Caffe: Open Source Deep Learning Library



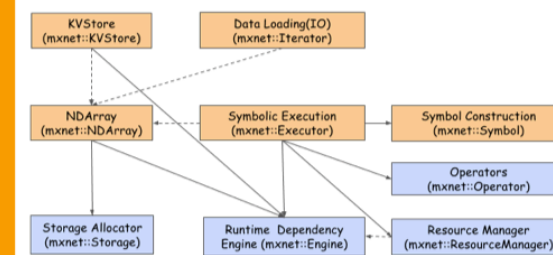
Maximally accurate	Maximally specific
espresso	0.88000
coffee	0.78000
beverage	0.80000
liquid	0.80000
fluid	0.80000



Tutorials by Evan Shelhamer, Jon Long, Sergio Guadarrama, Jeff Donahue, and Ross Girshick and Yangqing Jia



MXNet System Architecture



# PAI Tens

- Rich Data IO
- Distributed Job Optimization (Multi. GPU/CPU)
- Easy model Serving
- Hyper Parameter Tuning

```
odps@ test8>pai -name tensorflow -Dscript="file:///tmp/hello_tf.py";
ID = 20161209110921342g1j2sa
Sub Instance ID = 201612091909364c75f2fe_1457_4e93_bd96_05e63ac818e1
http://logview.odps.aliyun-inc.com:8080/logview/?h=http://100.81.182.94:54320/api&p=test8&i=201612091909364c75f2fe_1457_4e93_bd96_05e63ac818e1&token=TnJENnR2VU9RditsdXhQblRzY09GNETsemNRP5xPRFBTX09CTzoxMzY1OTM3MTUwNzcyMjE5LDE0E00DE40DY1ODIsey3TdGF0ZWl1bnQ1olt7IkFjdGlvbiI6WyJvZHBzOlJlYWQiXSwiRmZmZWNOIjoiQWxsY24iLCJ3ZXNvdXJjZSI6WyJhY3M6b2RwczoqOnByb2p1Y3RzL3Rlc3Q4L2luc3RhbmNlc3Y8MDE2MTIwOTE5MDkzNjRjNzVmMmZlXzE0NTdfNGU5M19iZDk2XzA1ZTYzYW44MThMSjdfV0sIlZlcnNpb24iOiIuIn0=
train: 2016-12-09 19:09:42 TensorflowTask_job:0/0/0[0%]
train: 2016-12-09 19:09:47 TensorflowTask_job:0/0/0[0%]
train: 2016-12-09 19:09:53 TensorflowTask_job:1/0/1[0%]
train: 2016-12-09 19:09:58 TensorflowTask_job:1/0/1[0%]
train: running
OK
```

**Main Content**

**Fuxi Jobs** | Summary | JSONSummary

**Fuxi Job Name:** algo\_test\_20161209

TaskName	Fatal/Finished/T
1 TensorflowTask	0//1
2 ps	0//1
3 worker	0//2

**Logview [Stdout]**

Only tail 10KB is shown here. Please click the button at bottom to download the whole log (max 5MB).

```

-----
alidate every 100 runs)
[worker:1] started training (validate every 100 runs)
[worker:1] step=0, global_step=0, loss=6.910646, duration=26.382059s.
[worker:1] step=1, global_step=1, loss=6.915818, duration=4.066880s.
[worker:1] step=2, global_step=2, loss=9.913230, duration=6.239480s.
[worker:0] step=0, global_step=0, loss=6.917369, duration=42.647304s.
[worker:0] step=1, global_step=2, loss=6.951333, duration=1.755433s.
[worker:0] step=2, global_step=2, loss=6.970006, duration=3.416824s.
[worker:1] step=3, global_step=1, loss=7.496880, duration=26.735484s.
[worker:0] step=3, global_step=3, loss=6.977772, duration=3.645903s.
[worker:1] step=4, global_step=3, loss=6.938528, duration=3.630591s.
[worker:0] step=4, global_step=4, loss=6.963563, duration=3.413888s.
[worker:1] step=5, global_step=4, loss=6.942375, duration=6.284963s.
[worker:0] step=5, global_step=4, loss=6.920757, duration=3.404328s.
[worker:1] step=6, global_step=5, loss=6.941708, duration=3.692321s.
[worker:0] step=6, global_step=5, loss=6.905121, duration=3.742672s.
[worker:1] step=7, global_step=6, loss=6.964681, duration=3.971193s.
[worker:0] step=7, global_step=6, loss=6.962502, duration=3.952058s.
  
```





# Pluto

A decorative graphic on the left side of the slide, consisting of a series of concentric, semi-circular lines that form a funnel-like shape pointing towards the right.

# Single-card Optimization

- Compiler-oriented strategy
  - Fuse small ops into bigger one
    - Reduce CUDA kernel launch overhead
  - Prepare data layout friendly with low-level computation library
- Memory optimization
  - Here again compiler-oriented tactics
    - Dependency analysis
    - Lifetime analysis

# Multi-cards Optimization

- **Heuristic-based Model Parallelism**
  - Both model weights and feature map taken into consideration
  - Memory allocator strategy taken into consideration
  - A greedy allocation algorithm
    - With pre-run support



## Multi-cards Optimization

- Hybrid-parallelism
  - Mixture of data-parallelism and model-parallelism
  - For communication-intensive parts, consider model-parallelism
  - For computation-intensive parts, consider data-parallelism
  - Tricks
    - Integrate seamlessly with computation graph style
    - Happier with pyramid network

# Multi-cards Optimization

- Hybrid-parallelism(cont.)

	Data-parallelism			
	1 GPU	2 GPU's	4 GPU's	8 GPU's
Data size(MiB)	0	930	1861	3722
$T_{fake}$ (s)	2.297	2.906	3.672	5.466
Speed-up	1X	0.79X	0.63X	0.42X
$T_{real}$ (s)	2.615	2.861	3.574	5.882
Speed-up	1X	0.91X	0.73X	0.44X
	Hybrid-parallelism			
	1 GPU	2 GPU's	4 GPU's	8 GPU's
Data size(MiB)	0	54	89	161
$T_{fake}$ (s)	2.297	1.238	0.662	0.443
Speed-up	1X	1.86X	3.47X	5.19X
$T_{real}$ (s)	2.615	1.500	0.831	0.552
Speed-up	1X	1.74X	3.15X	4.74X

M40 Result

	Data-parallelism			
	1 GPU	2 GPU's	4 GPU's	8 GPU's
$T_{fake}$ (s)	2.95	3.517	5.737	7.169
Speed-up	1X	0.84X	0.51X	0.41X
$T_{real}$ (s)	3.487	8.956	9.906	7.407
Speed-up	1X	0.39X	0.35X	0.47X
	Hybrid-parallelism			
	1 GPU	2 GPU's	4 GPU's	8 GPU's
$T_{fake}$ (s)	2.95	1.615	0.957	0.681
Speed-up	1X	1.83X	3.08X	4.33X
$T_{real}$ (s)	3.487	2.304	1.329	0.776
Speed-up	1X	1.51X	2.62X	4.49X

K40 Result

# Multi-cards Optimization

- Late-multiply
  - Customized for fully-connected layers
  - Trade-off between computation and communication

**Algorithm 1** Distributed version of backward propagation.

**worker 0**

$$W_0 \leftarrow E_0^T X_0$$

Allreduce  $W_i$  to worker 0

$$W_{avg} \leftarrow \frac{\sum_i W_i}{n}$$

Broadcast  $W_{avg}$  to other workers

**other workers**

$$W_i \leftarrow E_i^T X_i$$

Allreduce  $W_i$  to worker 0

do nothing

Broadcast  $W_{avg}$  to other workers

$O(N * M)$



**Algorithm 2** Late Multiply on worker  $i$

Allgather  $X_i, E_i$

$$W_{avg} \leftarrow \frac{\sum_i E_i^T X_i}{n}$$

$O(N) + O(M)$

$W_{avg}: [N_l, N_{l+1}]$ ,  $X: [M, N_l]$ ,  $E: [M, N_{l+1}]$ , here  $N_l, N_{l+1}$  layer sizes,  $M$  is mini-batch size

# Multi-cards Optimization

- Late-multiply(cont.)

	10 Gbps Ethernet		
	1 GPU	2 GPUs	4 GPUs
$T$ (ms)	0	894	1287
$T_{opt}$ (ms)	0	164	399

	56 Gbps InfiniBand		
	1 GPU	2 GPUs	4 GPUs
$T$ (ms)	0	90	185
$T_{opt}$ (ms)	0	51	136

A decorative graphic element on the left side of the slide, consisting of a series of thin, curved lines that form a fan-like shape, pointing towards the title.

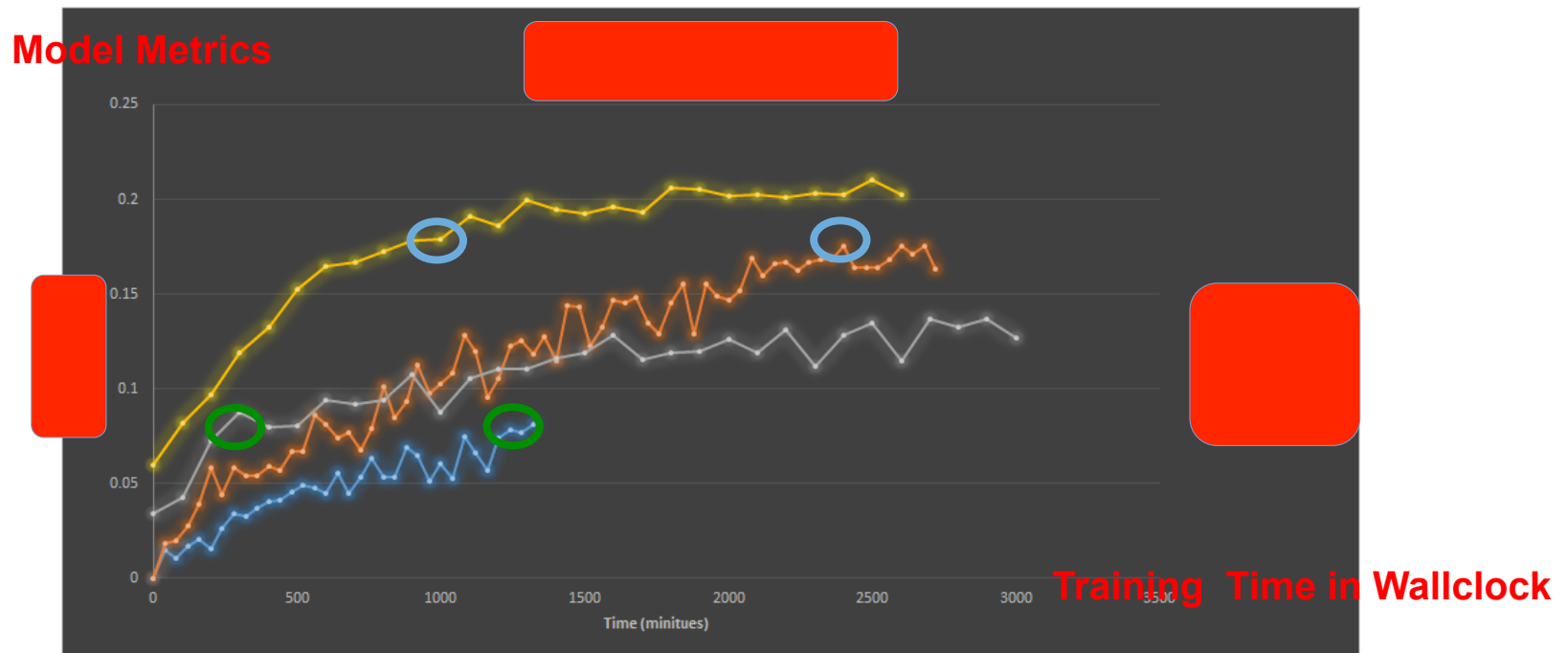
## Multi-cards Optimization

- Heuristic-based MA
  - Automatic batch-size selection
  - Learning rate auto-tuning
  - Happier with sequential model



# Multi-cards Optimization

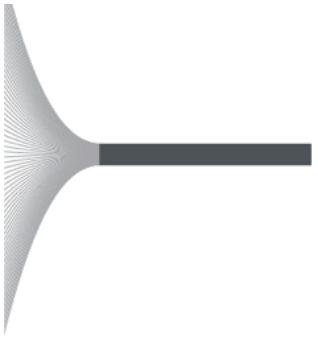
- Heuristic-based MA(cont.)



A decorative graphic on the left side of the slide, consisting of a series of concentric, semi-circular lines that form a funnel-like shape pointing towards the right.

# Inference Optimization

- Quantization
  - Significantly reduce model size(4X)
  - Around 2X speed-up on average
- Binarized Neural Network
  - Binarize model weights
  - Convert floating point computation into bit manipulation
  - Both model size and computation speed significantly improved
  - Training process needs to be manipulated to compensate for accuracy
  - Happier with CNN, but for RNN...

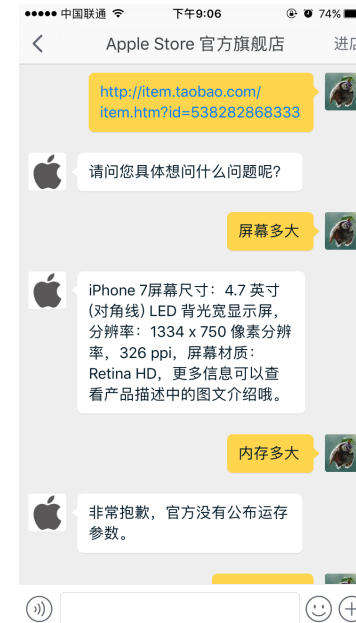


# PAI DL Application

# AliMe – Personal Assistant Bot in E-commerce



AliMe for  
Customers



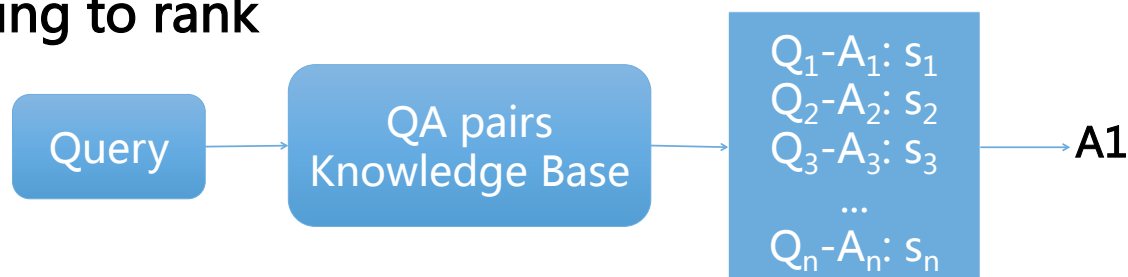
AliMe for  
Sellers



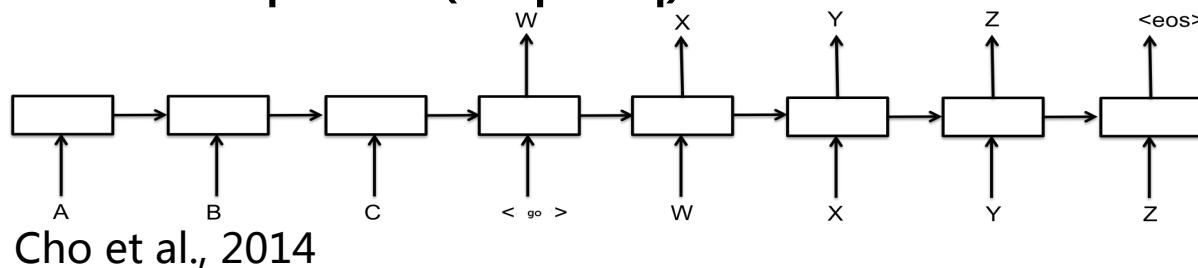
AliMe for  
Enterprises

# Open-Domain Conversations

- Retrieval Model
  - Learning to rank



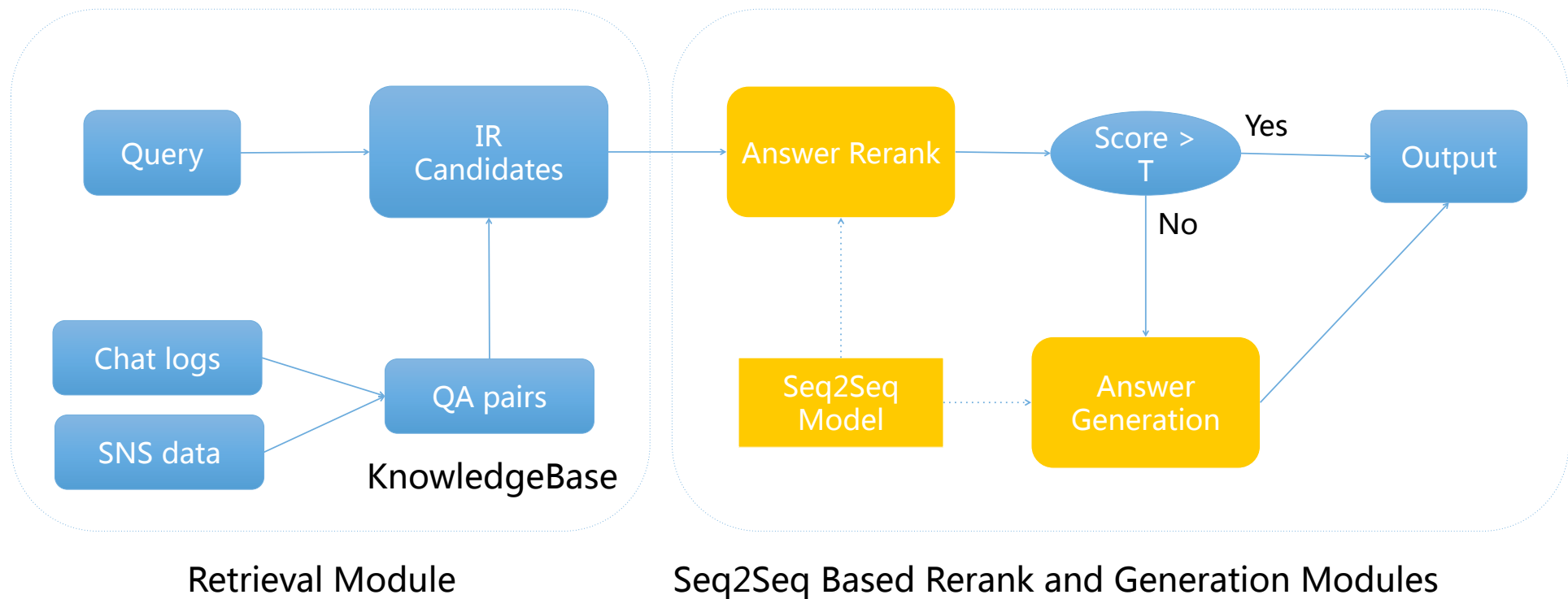
- Generation Model
  - Sequence to Sequence (Seq2Seq) Model



- Recurrent Neural Networks: LSTM, GRU (our choice)

# A Hybrid Conversation Model based on Seq2Seq

- Overview



A decorative graphic on the left side of the slide, consisting of a series of concentric, semi-circular lines that form a funnel-like shape pointing towards the right.

## PAI DL Support for AliMe

- Both the offline training and online serving backed by PAI
- Through heuristic-based MA, the offline training task has 2.8X convergence speed-up with 4 cards setting
- Through quantization, the online serving task has 1.5X speed-up on commodity CPU servers.

## Conclusion

- PAI DL
  - End2end machine learning platform
  - Support big data analytics
  - Optimized Deep learning algorithms
  - Scheduling on CPU / GPU cloud
  - More data intelligence...
- Pluto
  - Distributed optimization engine of PAI DL
- PAI DL Application
  - PAI DL makes it easy to build DL methods for industrial applications

数据智能 触手可及

SCAN BARCODE  
START YOUR TRIAL



Platform of  
Artificial Intelligence





# We are hiring! 😊

[muzhuo.yj@alibaba-inc.com](mailto:muzhuo.yj@alibaba-inc.com)

[chenyan.cy@alibaba-inc.com](mailto:chenyan.cy@alibaba-inc.com)

A decorative graphic on the left side of the slide, consisting of a series of concentric, semi-circular lines that form a funnel-like shape pointing towards the right.

## Reference

- AliMe Chat: A Sequence to Sequence and Rerank based Chatbot Engine, Minghui Qiu et al., ACL 2017.
- Deep Learning with PAI: a Case Study of AliMe, Minghui Qiu et al., Deep Learning Summit 2017.
- TensorFlow in AliMe, Jun Yang et al., Shanghai GDG Mar., 2017.

# Thanks!

MORE THAN JUST CLOUD |  Alibaba Cloud

