GIAC BEIJING Dec.12.16-17

架構 ARCHNOTES 高可用架构

小米深度学习平台架构与实践

技术架构未来

thegiac.com

About me

- 陈迪豪，就职小米，深度学习领域
- 分布式存储 - HBase - Ceph
- 基础架构 - Docker - OpenStack
- 人工智能 - TensorFlow - K8S
- Seagull作者，1400+ stars

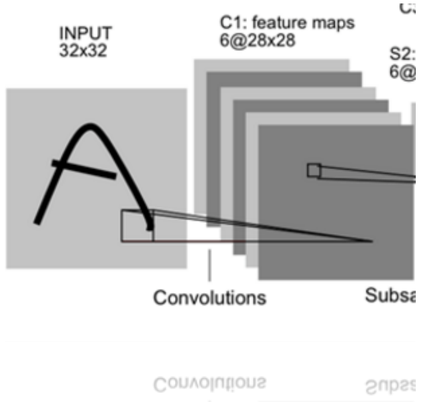
Agenda

- 机器学习与深度学习应用
- 深度学习平台架构与设计
- 深度学习平台应用与实践

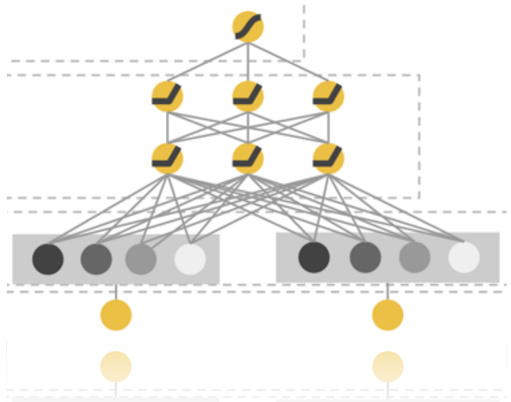
Agenda

- 机器学习与深度学习应用
- 深度学习平台架构与设计
- 深度学习平台应用与实践

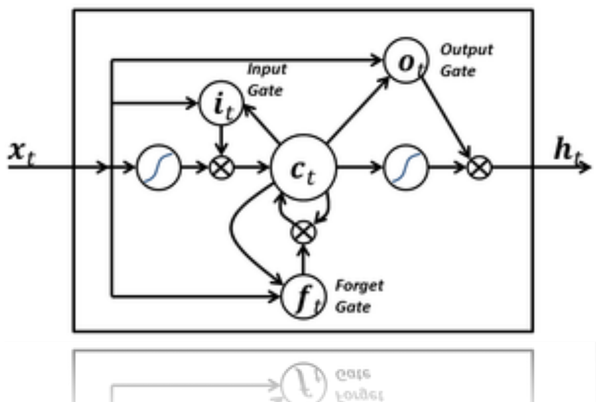
Introduce ML/DL



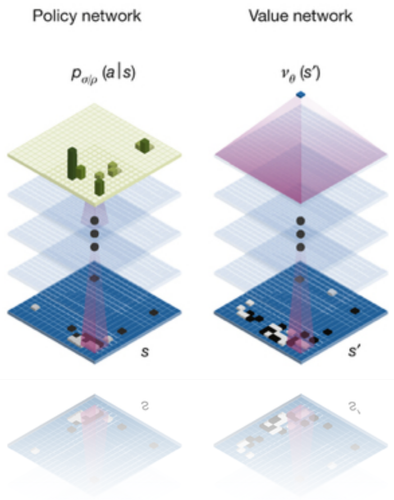
CNN



MLP

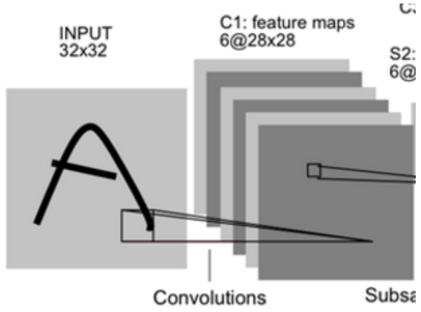


RNN/LSTM

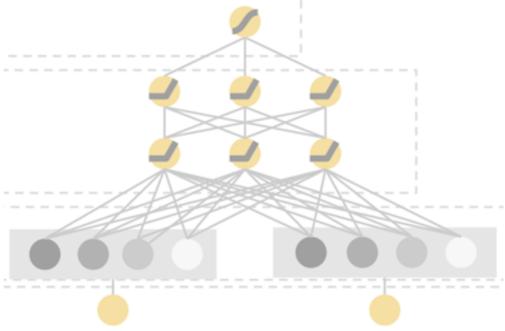


RL

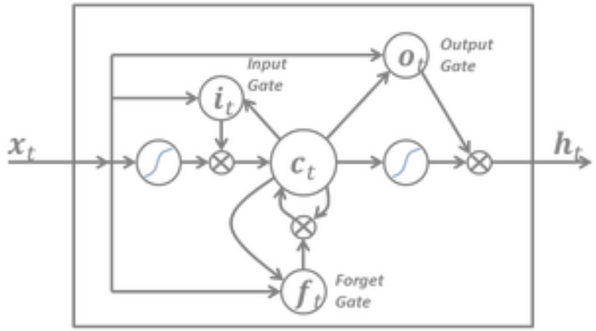
Introduce ML/DL



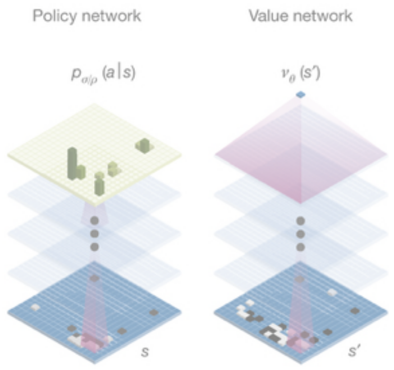
CNN



MLP

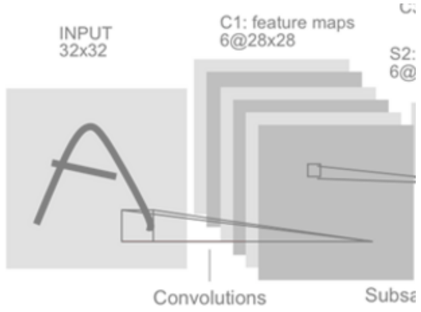


RNN/LSTM

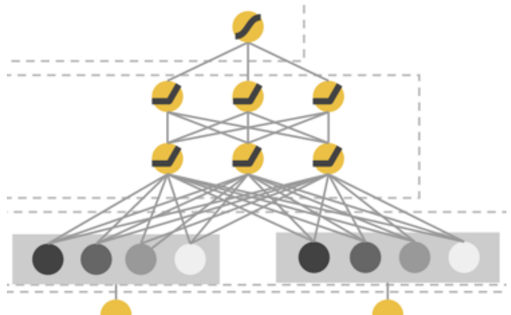


RL

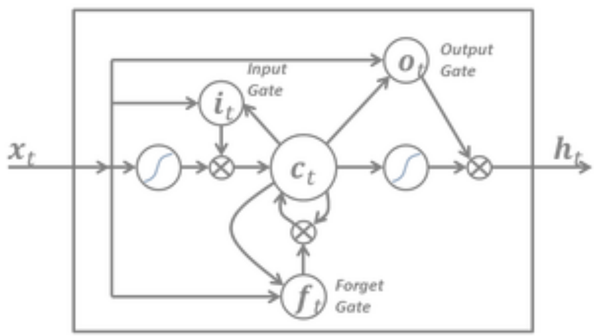
Introduce ML/DL



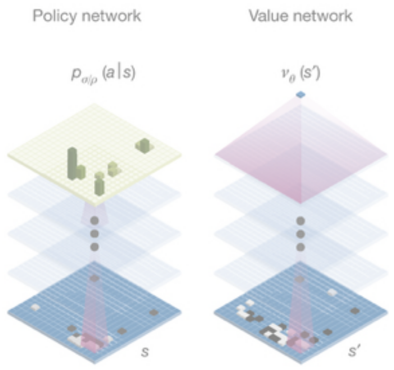
CNN



MLP

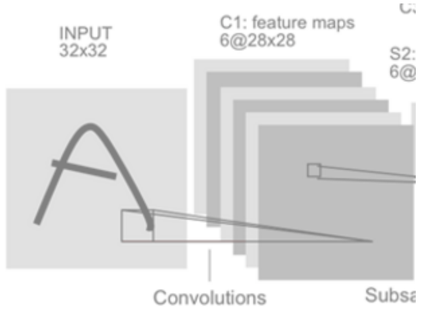


RNN/LSTM

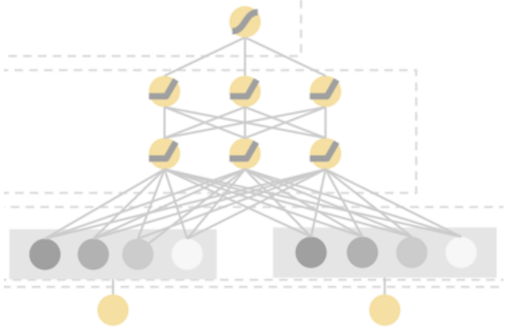


RL

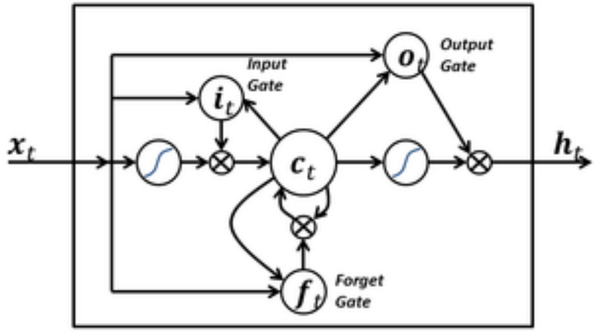
Introduce ML/DL



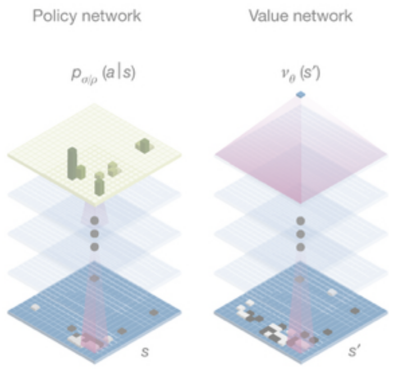
CNN



MLP

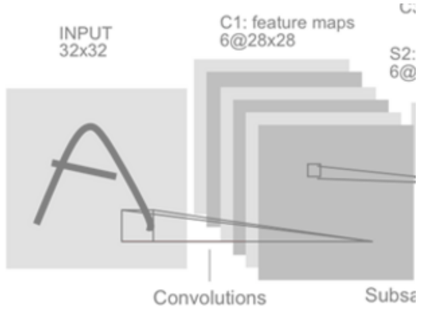


RNN/LSTM

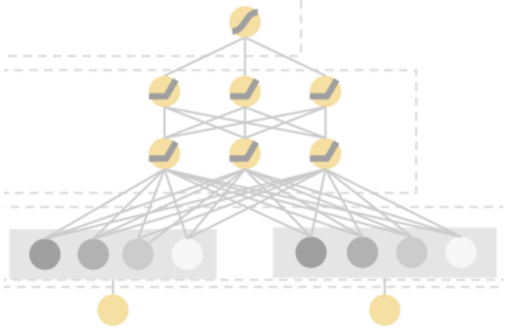


RL

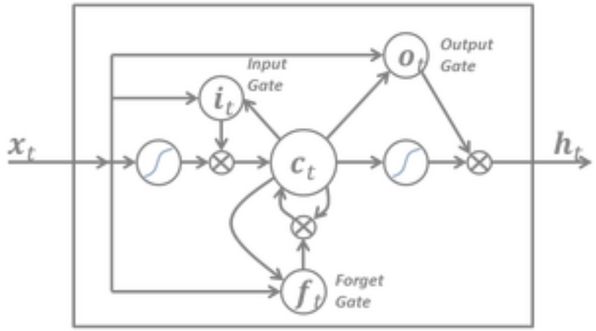
Introduce ML/DL



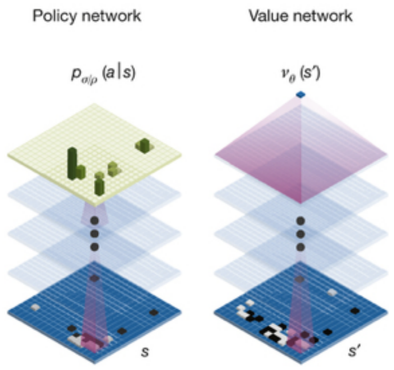
CNN



MLP

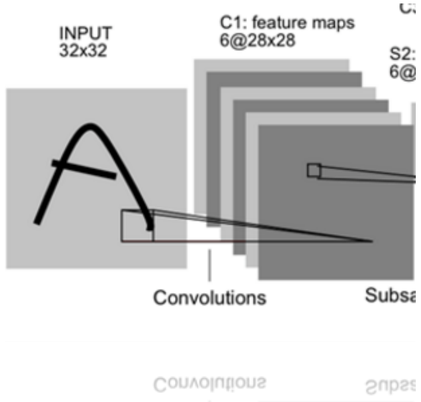


RNN/LSTM

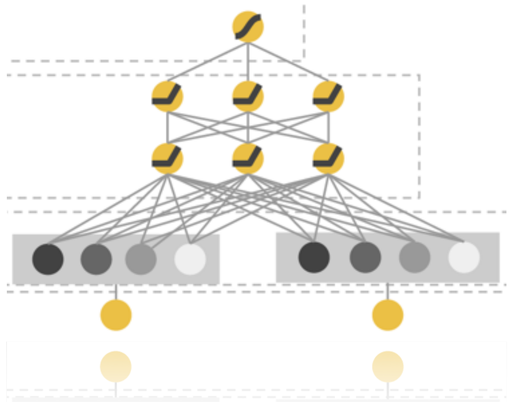


RL

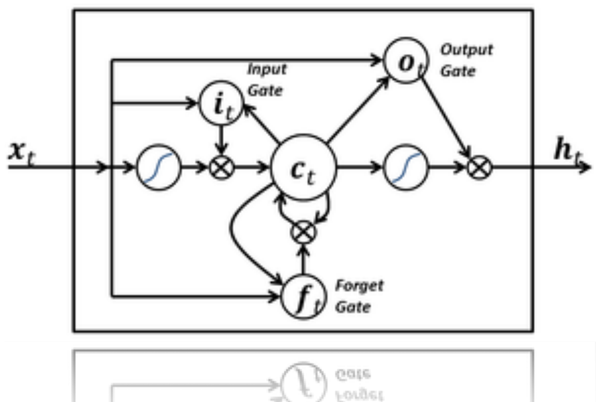
Introduce ML/DL



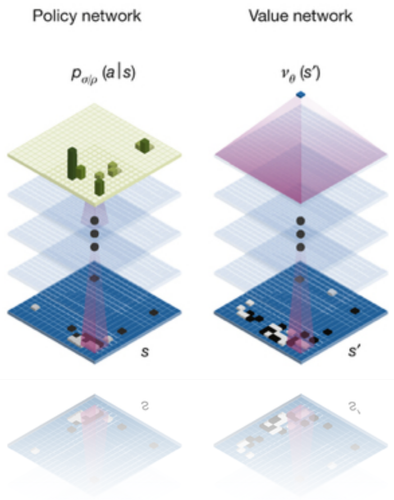
CNN



MLP



RNN/LSTM

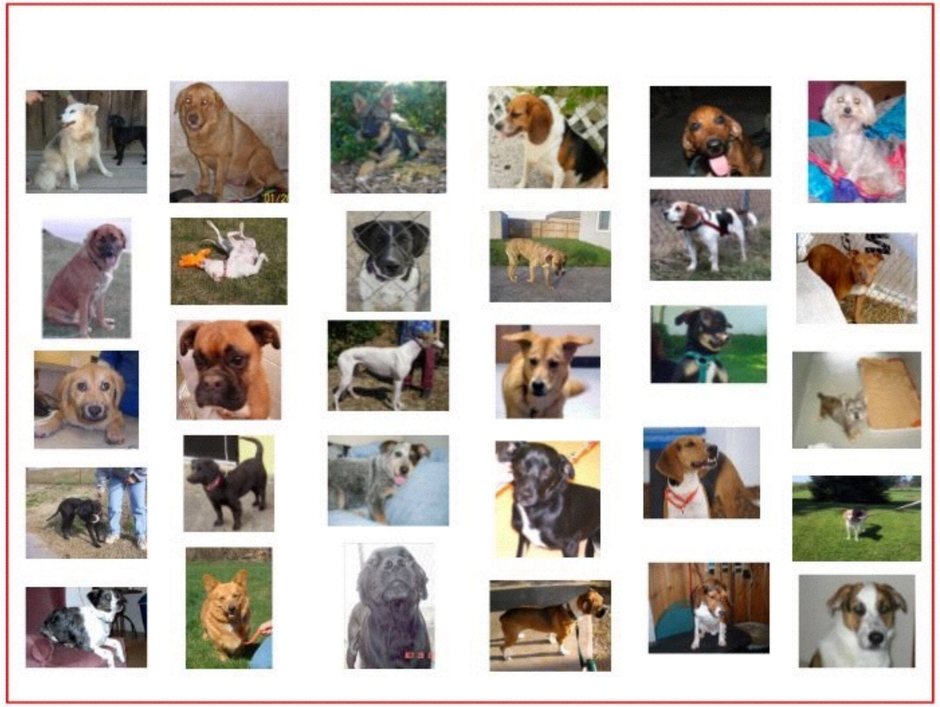


RL

Case Study: Image Classification

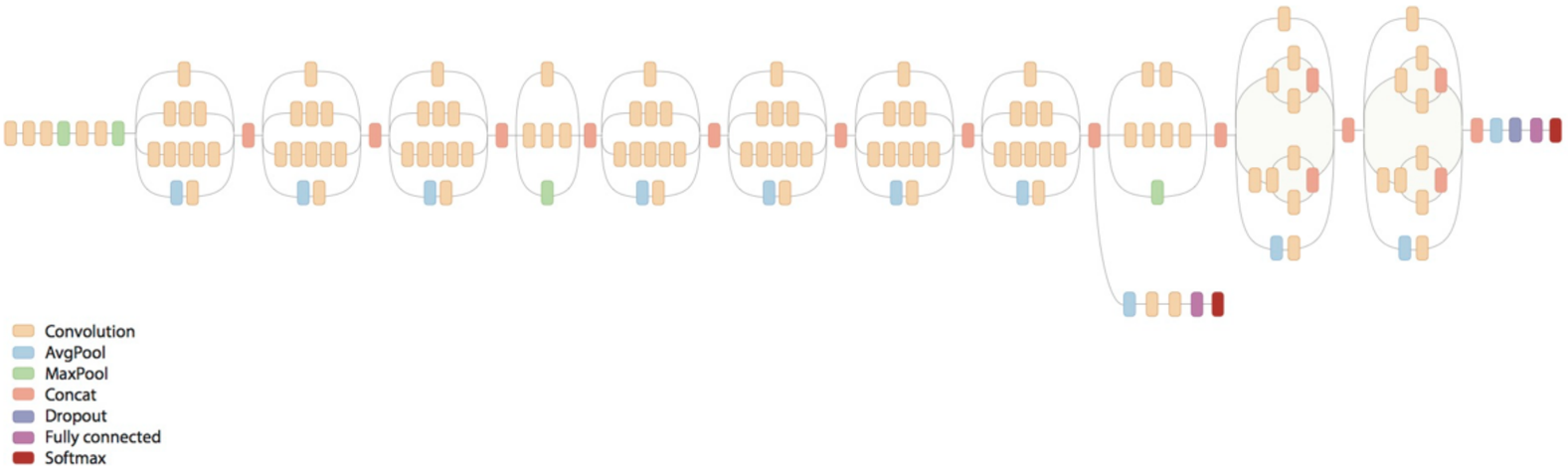
Cats

Dogs

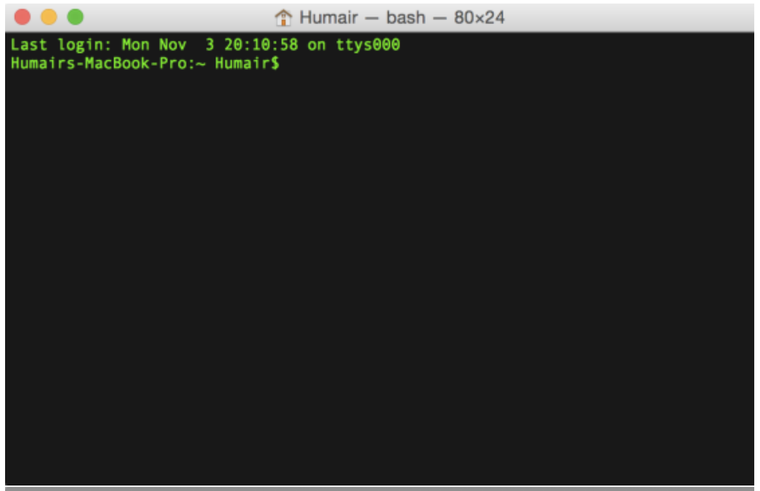


Sample of cats & dogs images from Kaggle Dataset

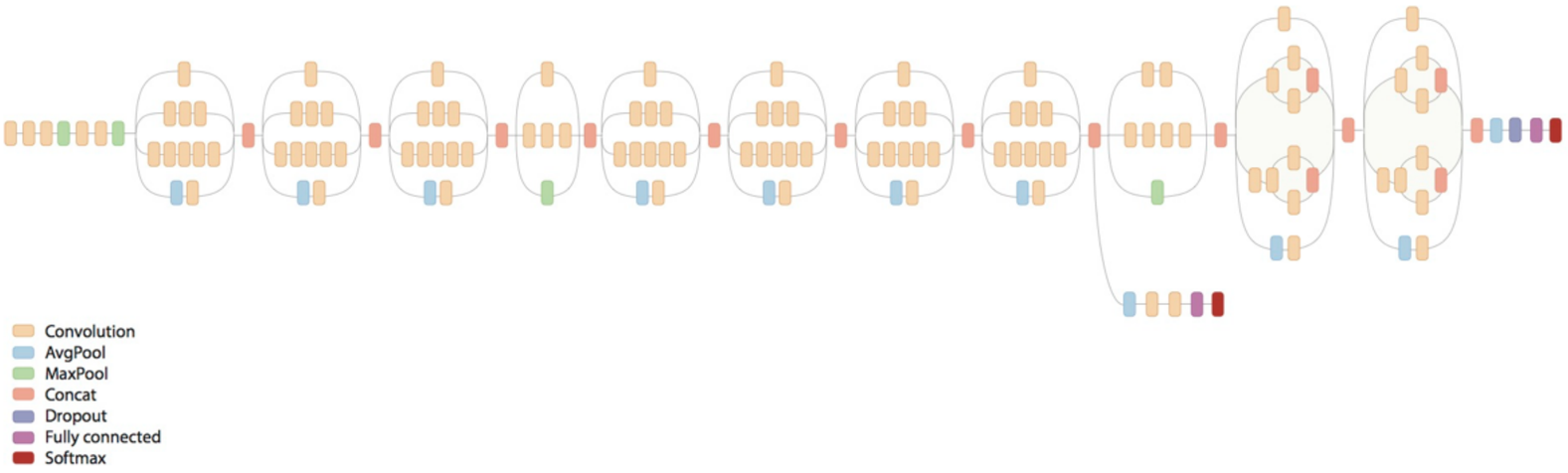
Case Study: Image Classification



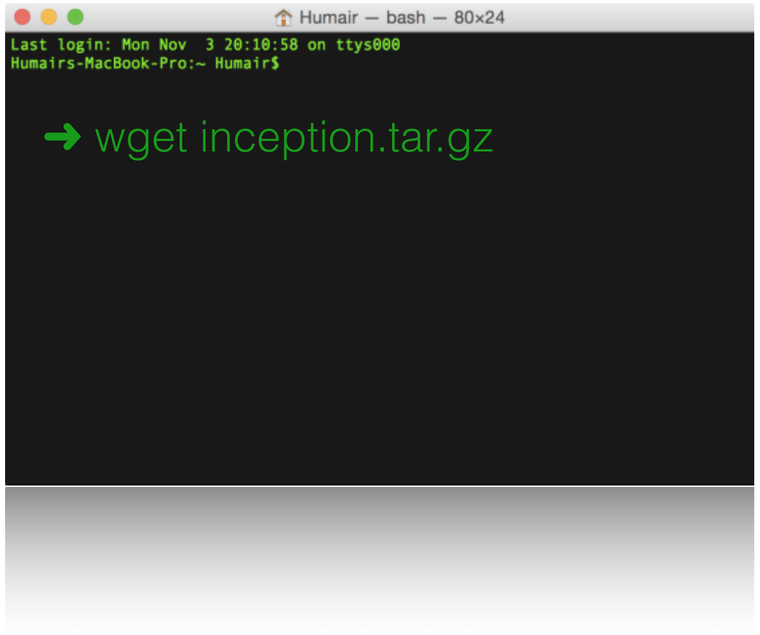
https://github.com/tensorflow/models/tree/master/inception



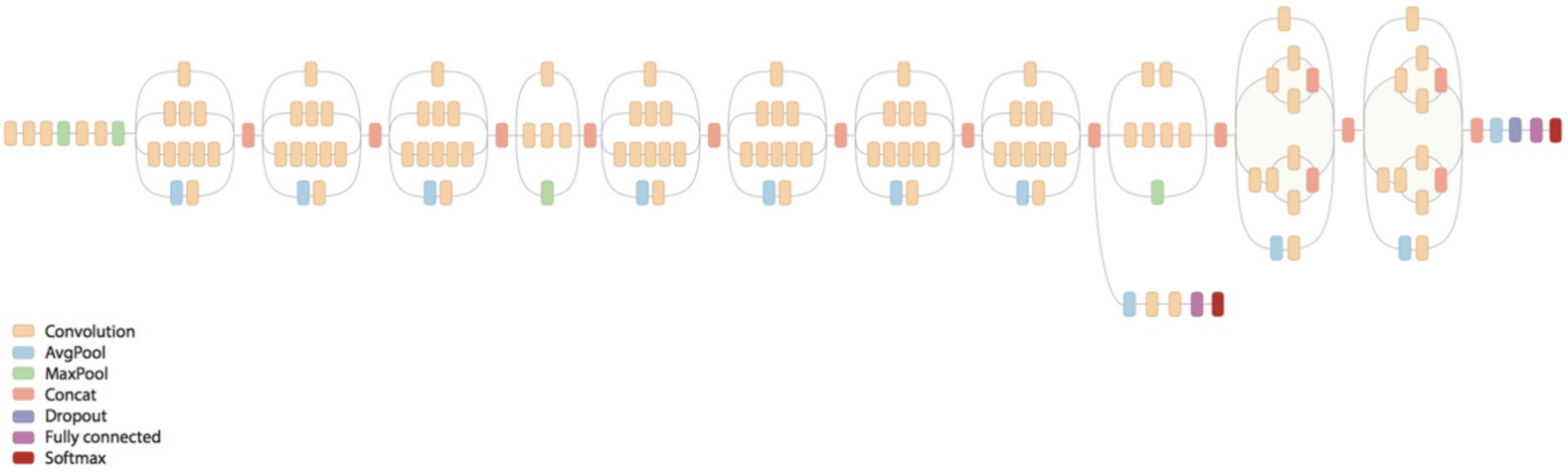
Case Study: Image Classification



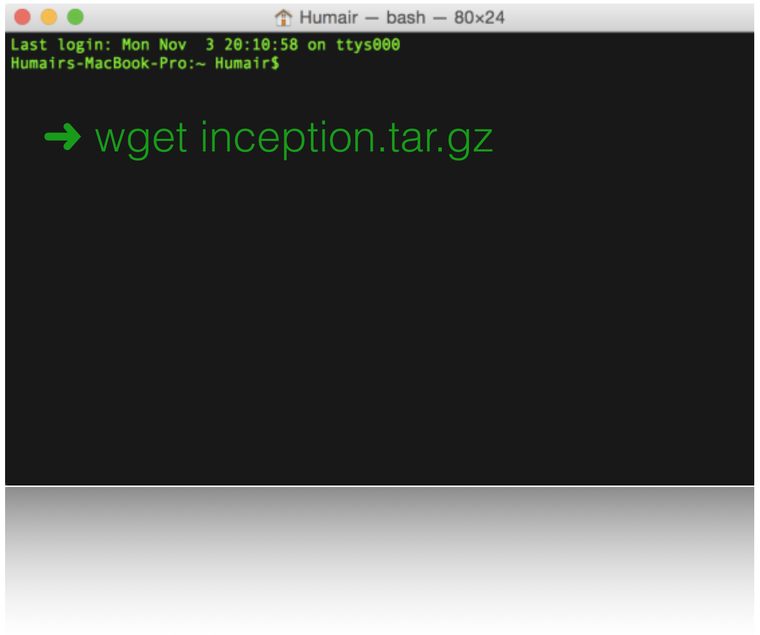
https://github.com/tensorflow/models/tree/master/inception



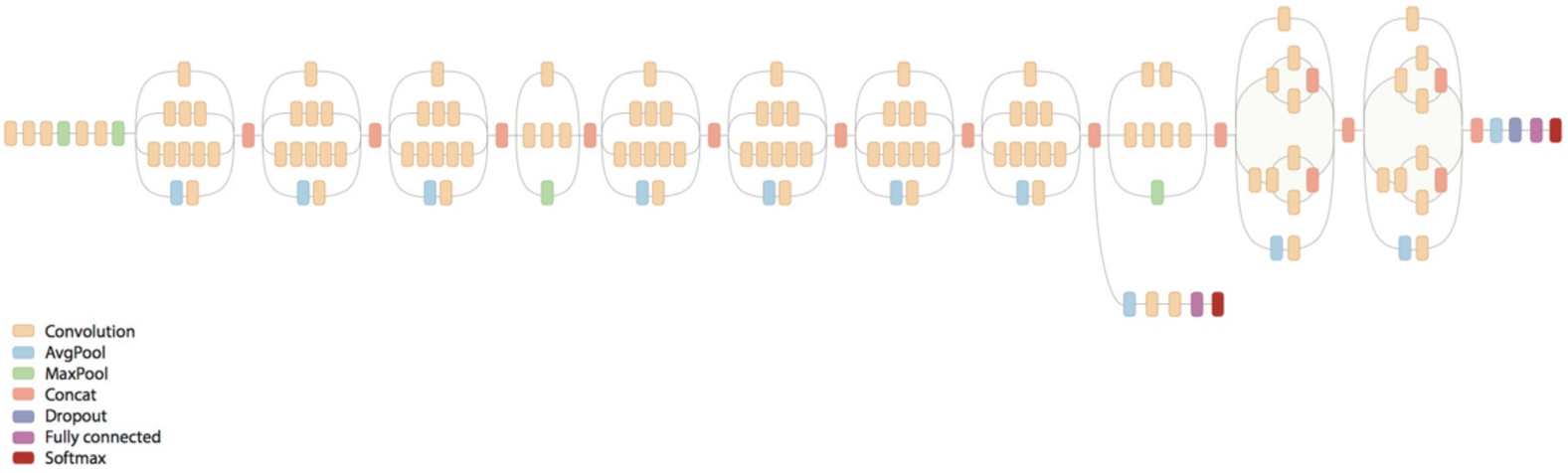
Case Study: Image Classification



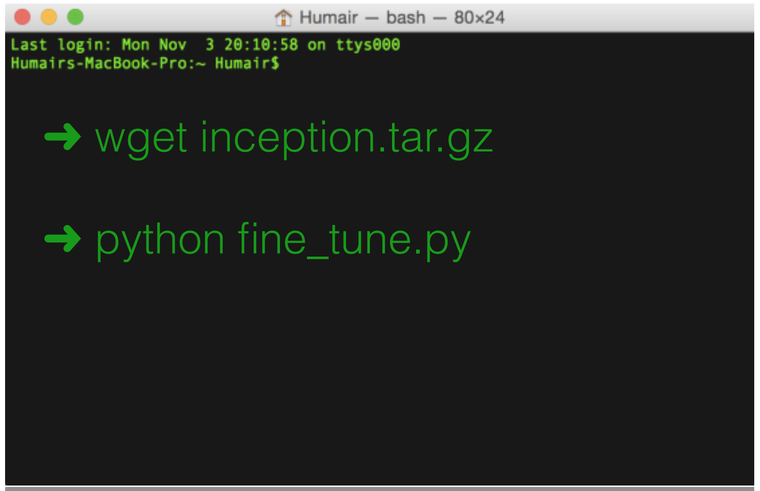
https://github.com/tensorflow/models/tree/master/inception



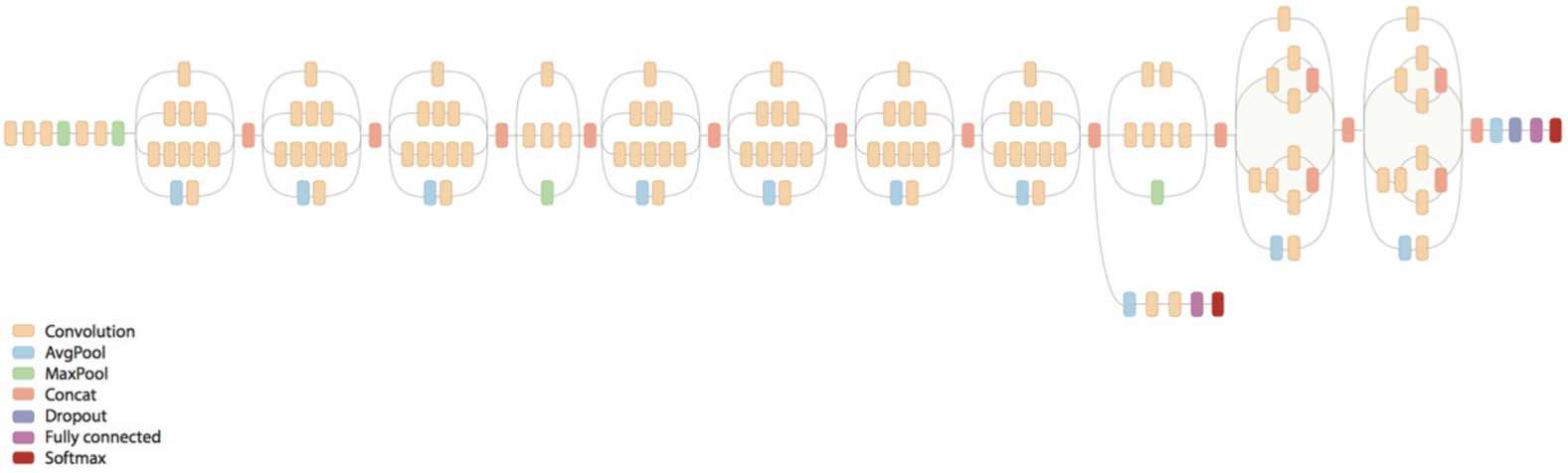
Case Study: Image Classification



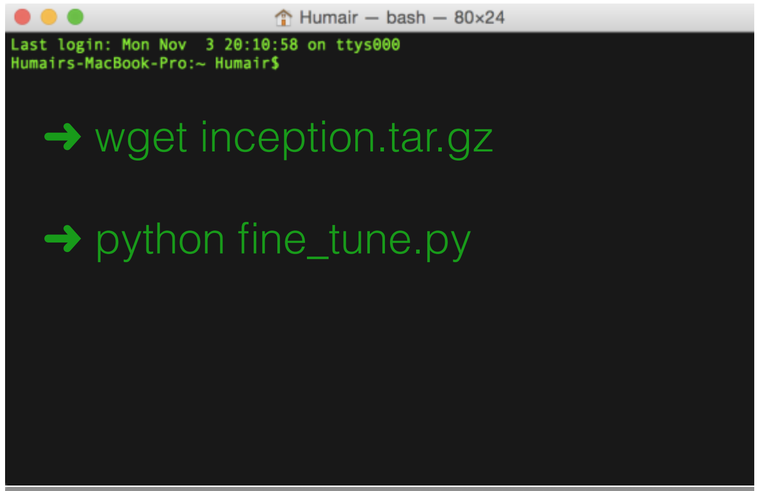
https://github.com/tensorflow/models/tree/master/inception



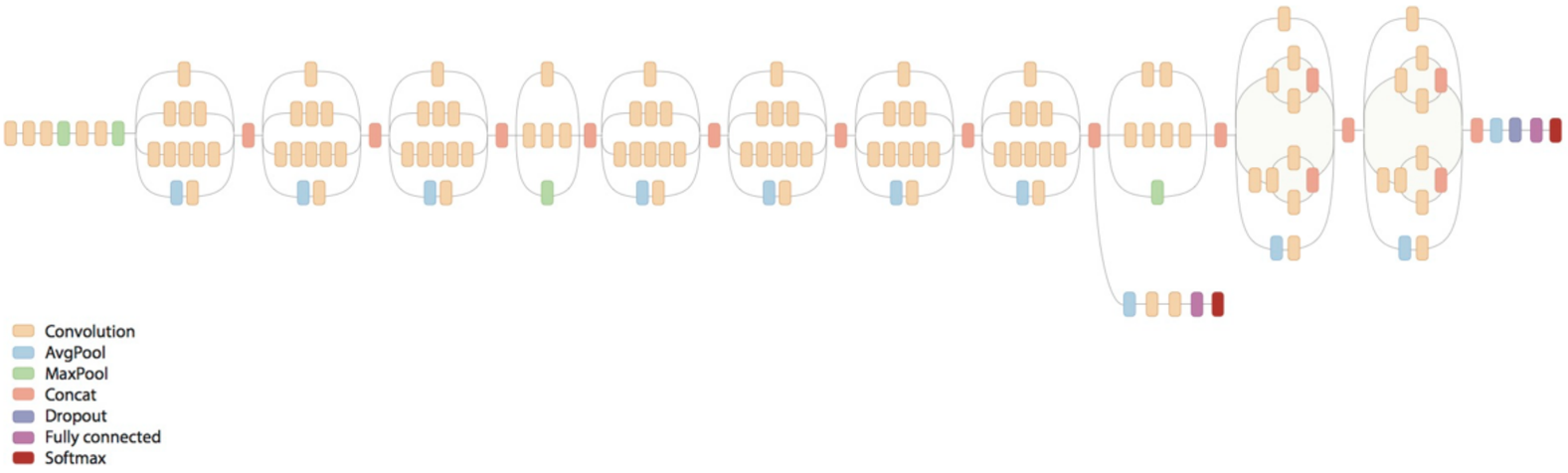
Case Study: Image Classification



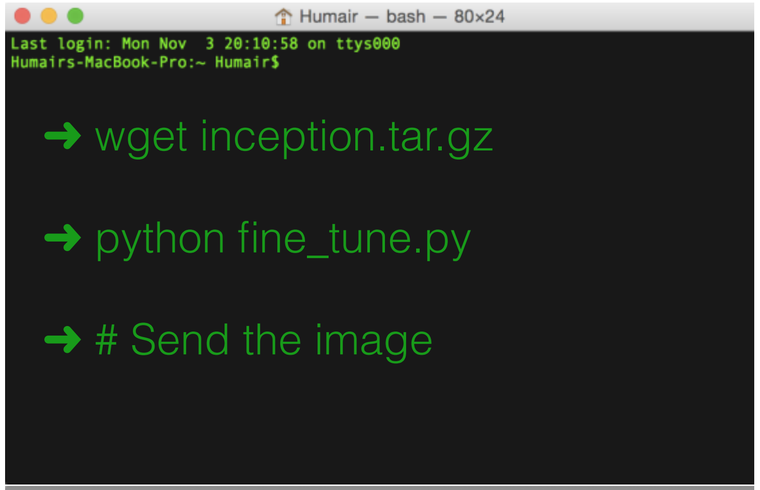
https://github.com/tensorflow/models/tree/master/inception



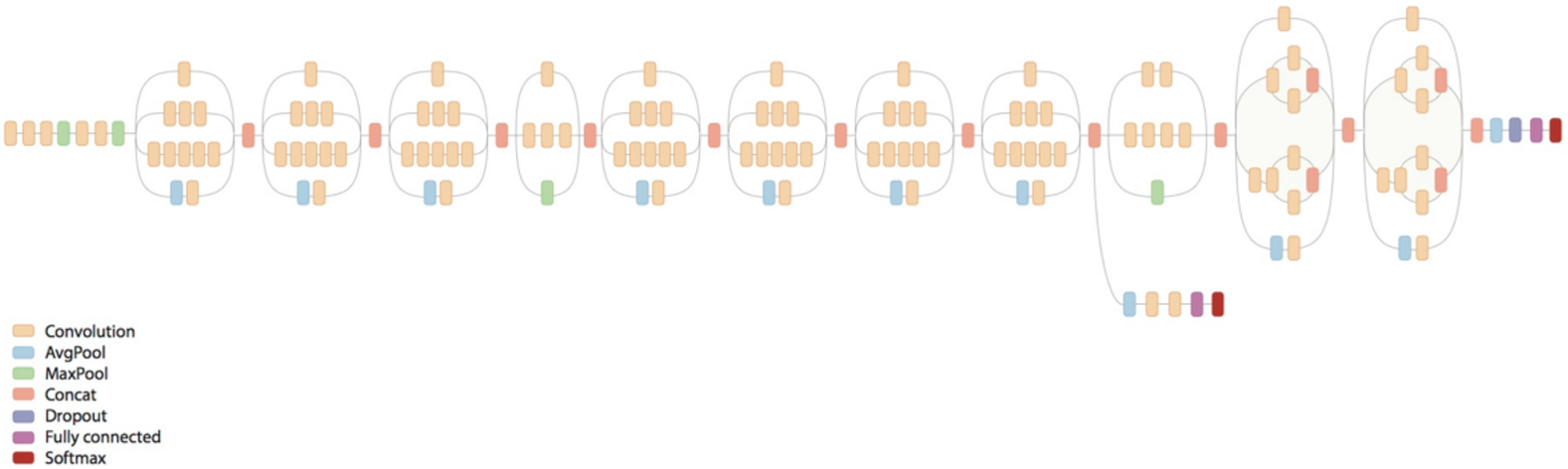
Case Study: Image Classification



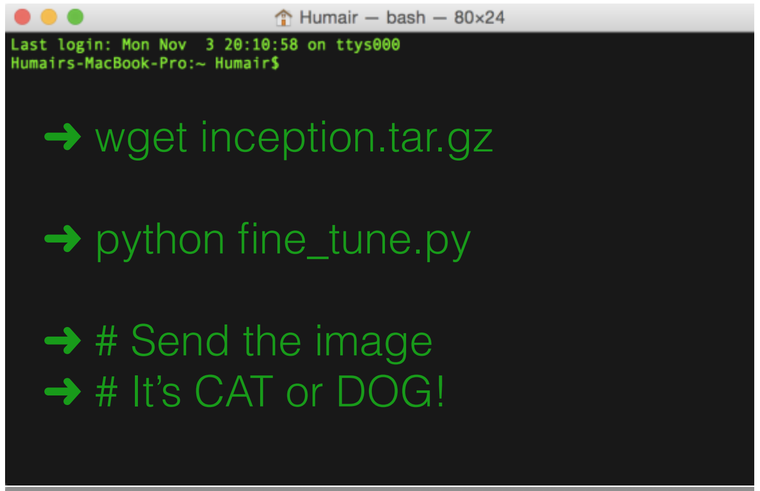
https://github.com/tensorflow/models/tree/master/inception



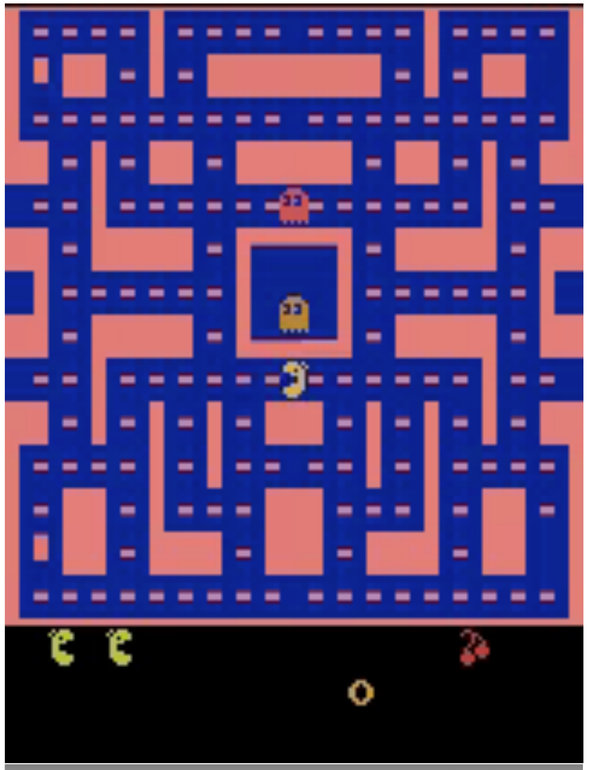
Case Study: Image Classification

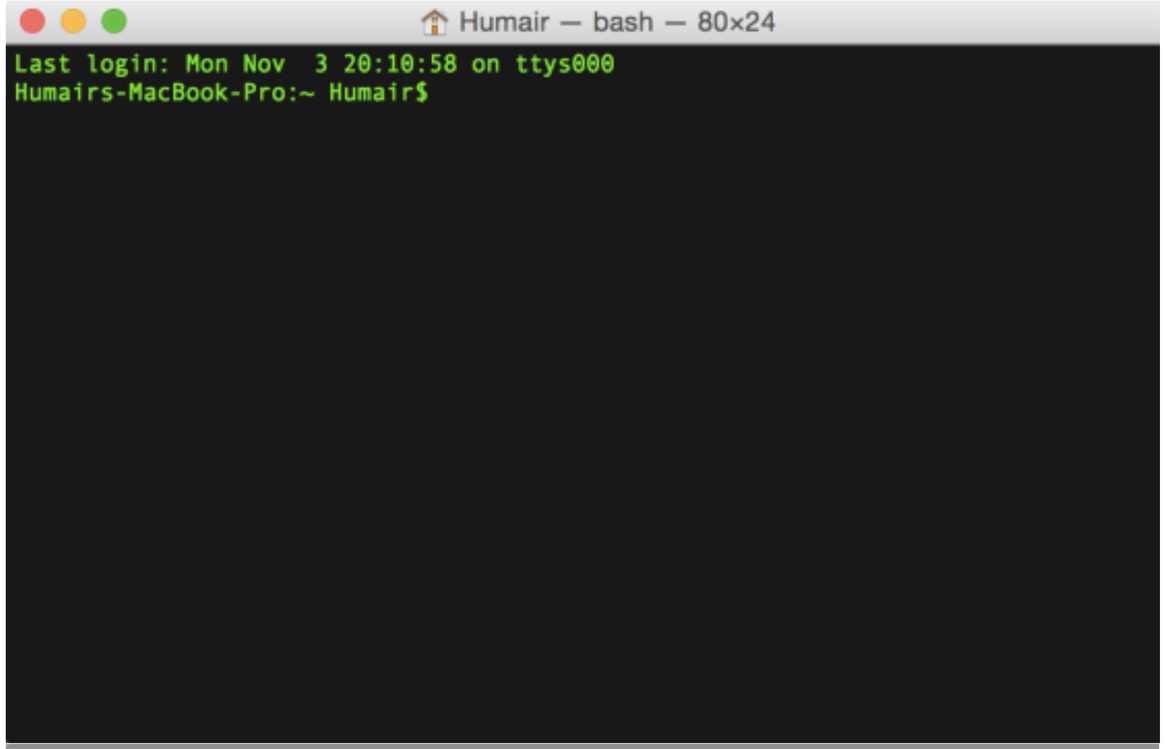


https://github.com/tensorflow/models/tree/master/inception

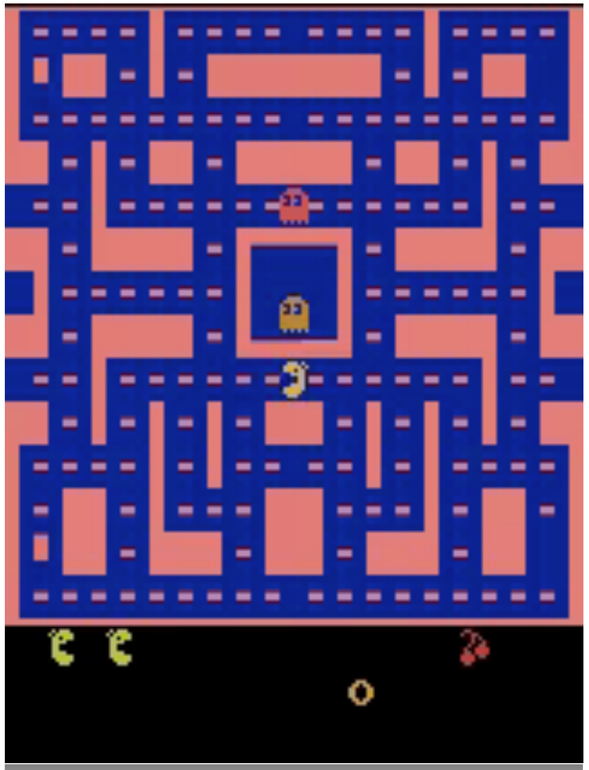


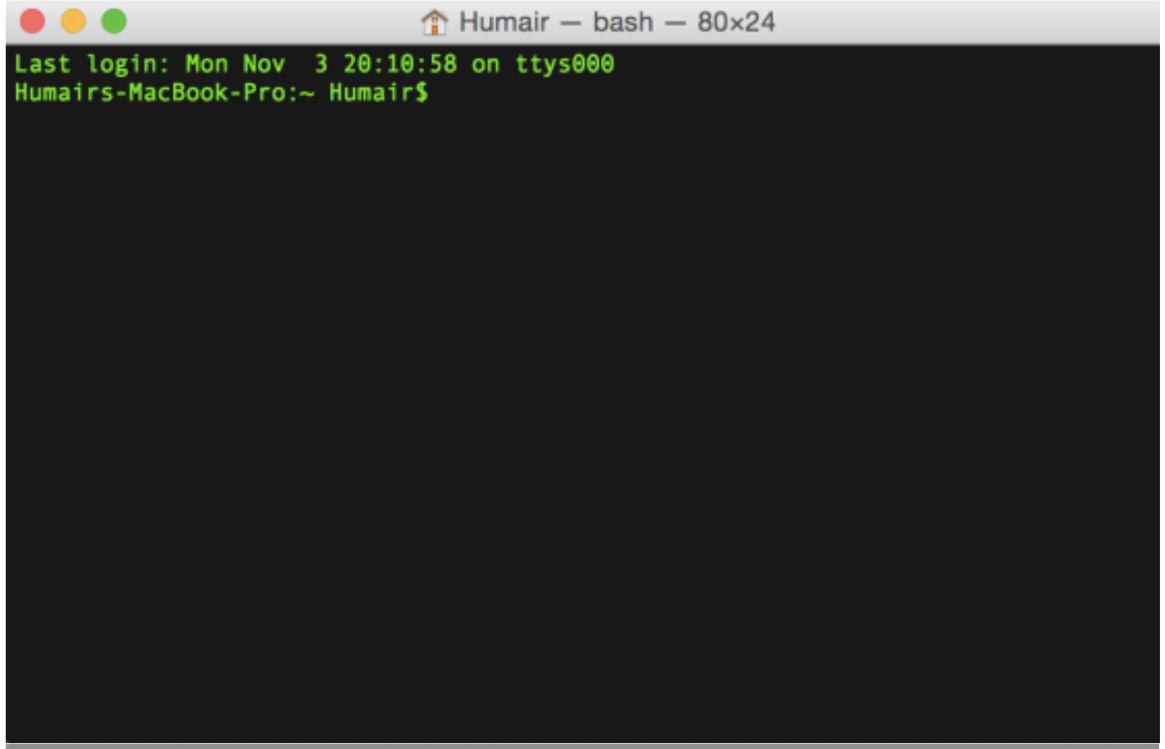
Case Study: Games AI



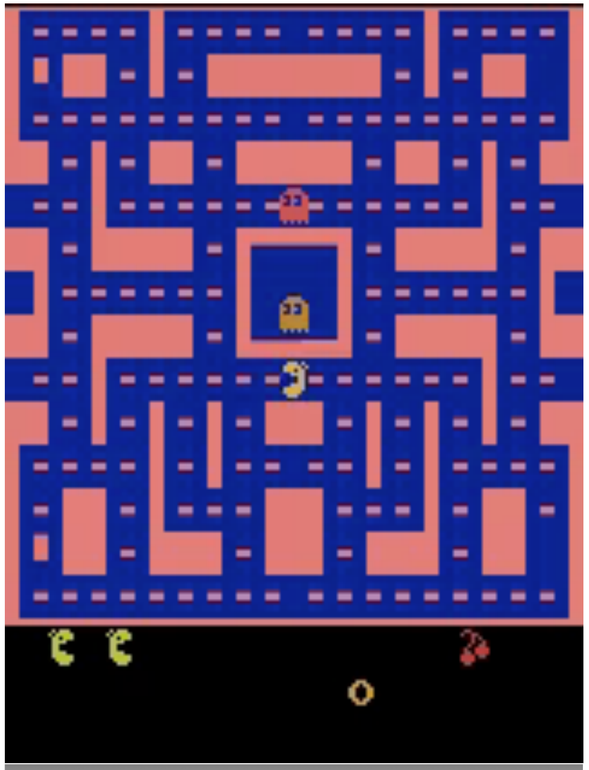


Case Study: Games AI





Case Study: Games AI



Humair — bash — 80×24

```
Last login: Mon Nov  3 20:10:58 on ttys000
Humairs-MacBook-Pro:~ Humair$

import gym

env = gym.make("MsPacman-v0")

observation = env.reset()

for _ in range(1000):
    env.render()
    action = env.action_space.sample()
    env.step(action)
```

Introduce TensorFlow

TensorFlow

GIAC | BEIJING Dec.12.16-17

thegiac.com

Introduce TensorFlow

- Deep learning library

Introduce TensorFlow

- Deep learning library
- C++/Python interfaces

Introduce TensorFlow

- Deep learning library
- C++/Python interfaces
- Training with CPU/GPU

Introduce TensorFlow

TensorFlow

- Deep learning library
- C++/Python interfaces
- Training with CPU/GPU
- Distributed for large scale

Introduce TensorFlow

```
# Import the library
import tensorflow as tf

# Define the graph
hello_op = tf.constant('Hello, TensorFlow!')
a = tf.constant(10)
b = tf.constant(32)
compute_op = tf.add(a, b)

# Define the session to run graph
with tf.Session() as sess:
    print(sess.run(hello_op))
    print(sess.run(compute_op))
```

Introduce TensorFlow

```python
# Import the library
import tensorflow as tf

# Define the graph
hello_op = tf.constant('Hello, TensorFlow!')
a = tf.constant(10)
b = tf.constant(32)
compute_op = tf.add(a, b)

# Define the session to run graph
with tf.Session() as sess:
    print(sess.run(hello_op))
    print(sess.run(compute_op))
```

Introduce TensorFlow

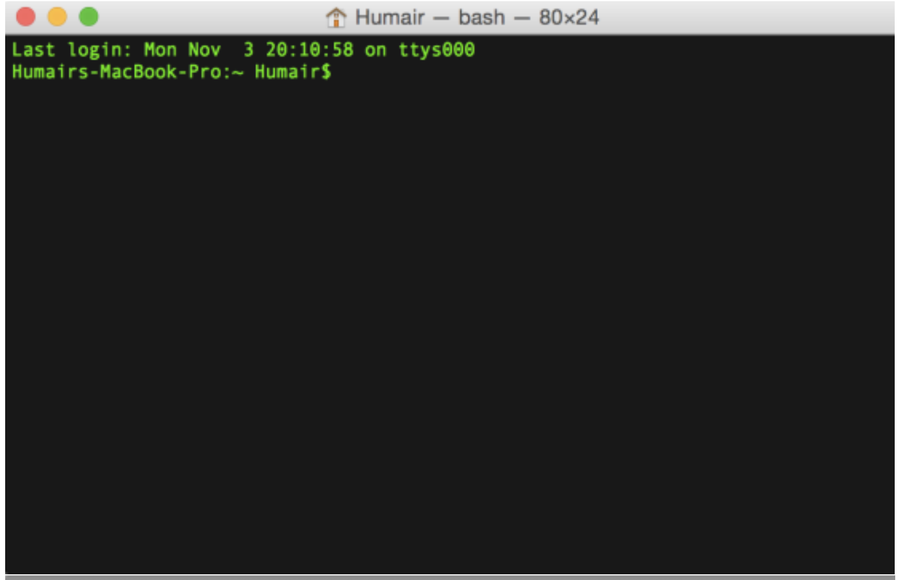
```
# Import the library
import tensorflow as tf

# Define the graph
hello_op = tf.constant('Hello, TensorFlow!')
a = tf.constant(10)
b = tf.constant(32)
compute_op = tf.add(a, b)

# Define the session to run graph
with tf.Session() as sess:
    print(sess.run(hello_op))
    print(sess.run(compute_op))
```

Introduce TensorFlow

```
# Import the library
import tensorflow as tf

# Define the graph
hello_op = tf.constant('Hello, TensorFlow!')
a = tf.constant(10)
b = tf.constant(32)
compute_op = tf.add(a, b)

# Define the session to run graph
with tf.Session() as sess:
    print(sess.run(hello_op))
    print(sess.run(compute_op))
```

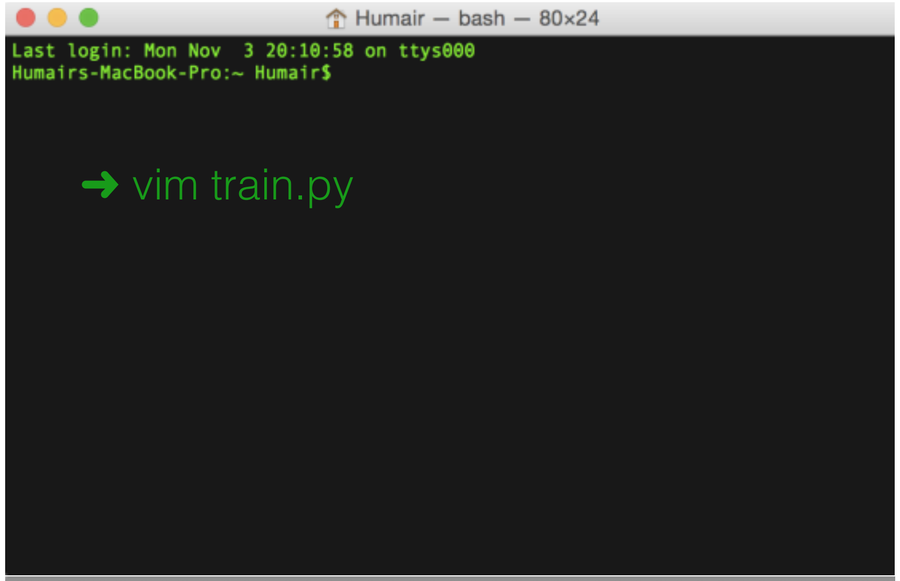
Introduce TensorFlow

```python
# Import the library
import tensorflow as tf

# Define the graph
hello_op = tf.constant('Hello, TensorFlow!')
a = tf.constant(10)
b = tf.constant(32)
compute_op = tf.add(a, b)

# Define the session to run graph
with tf.Session() as sess:
    print(sess.run(hello_op))
    print(sess.run(compute_op))
```



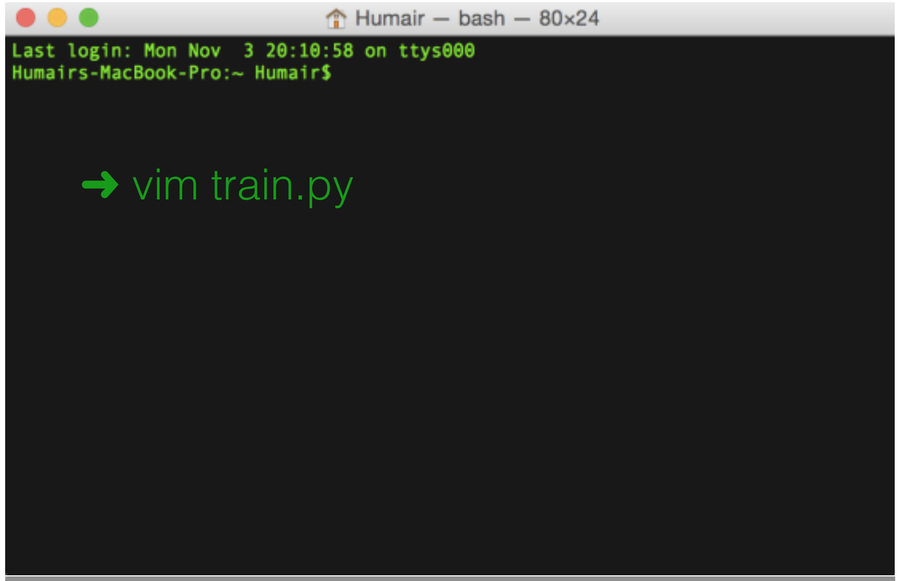
Introduce TensorFlow

```python
# Import the library
import tensorflow as tf

# Define the graph
hello_op = tf.constant('Hello, TensorFlow!')
a = tf.constant(10)
b = tf.constant(32)
compute_op = tf.add(a, b)

# Define the session to run graph
with tf.Session() as sess:
    print(sess.run(hello_op))
    print(sess.run(compute_op))
```



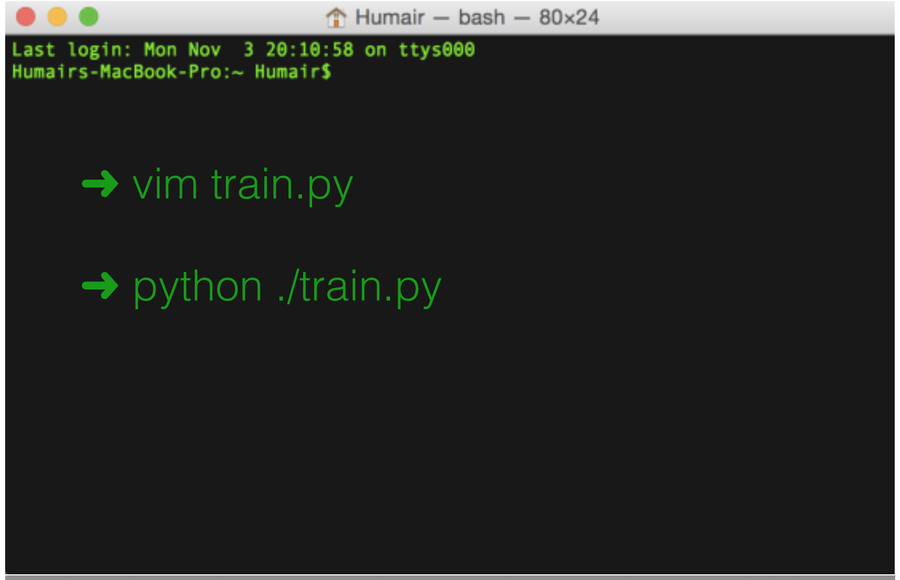
Introduce TensorFlow

```python
# Import the library
import tensorflow as tf

# Define the graph
hello_op = tf.constant('Hello, TensorFlow!')
a = tf.constant(10)
b = tf.constant(32)
compute_op = tf.add(a, b)

# Define the session to run graph
with tf.Session() as sess:
    print(sess.run(hello_op))
    print(sess.run(compute_op))
```



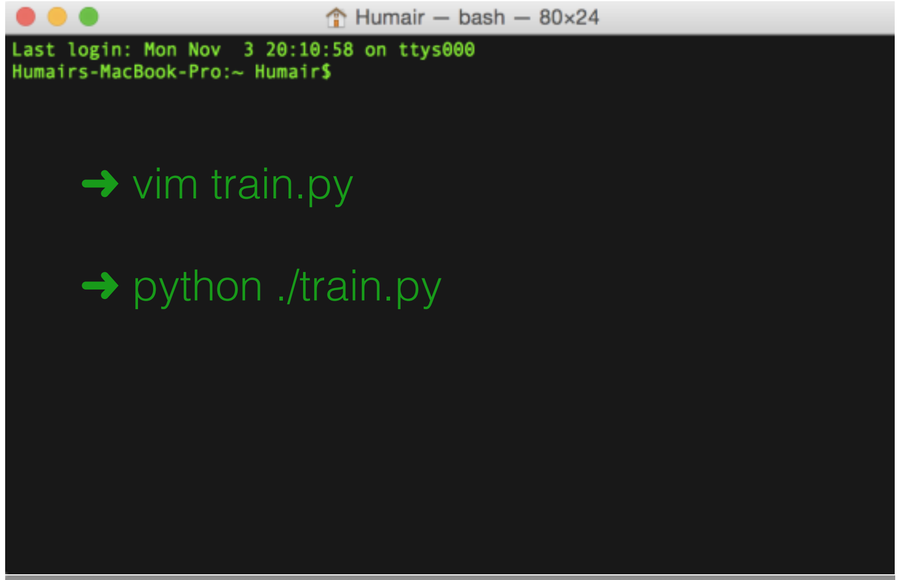
Introduce TensorFlow

```python
# Import the library
import tensorflow as tf

# Define the graph
hello_op = tf.constant('Hello, TensorFlow!')
a = tf.constant(10)
b = tf.constant(32)
compute_op = tf.add(a, b)

# Define the session to run graph
with tf.Session() as sess:
    print(sess.run(hello_op))
    print(sess.run(compute_op))
```



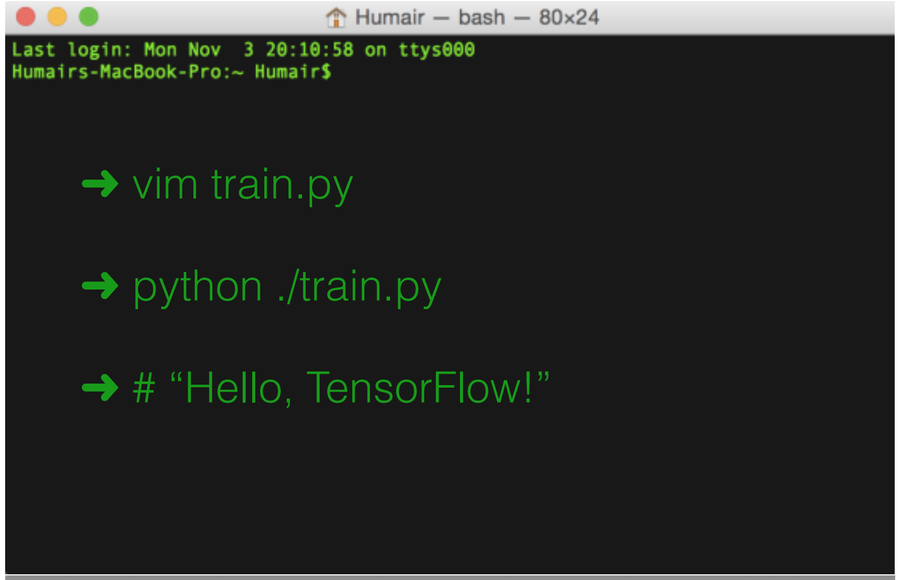
Introduce TensorFlow

```python
# Import the library
import tensorflow as tf

# Define the graph
hello_op = tf.constant('Hello, TensorFlow!')
a = tf.constant(10)
b = tf.constant(32)
compute_op = tf.add(a, b)

# Define the session to run graph
with tf.Session() as sess:
    print(sess.run(hello_op))
    print(sess.run(compute_op))
```



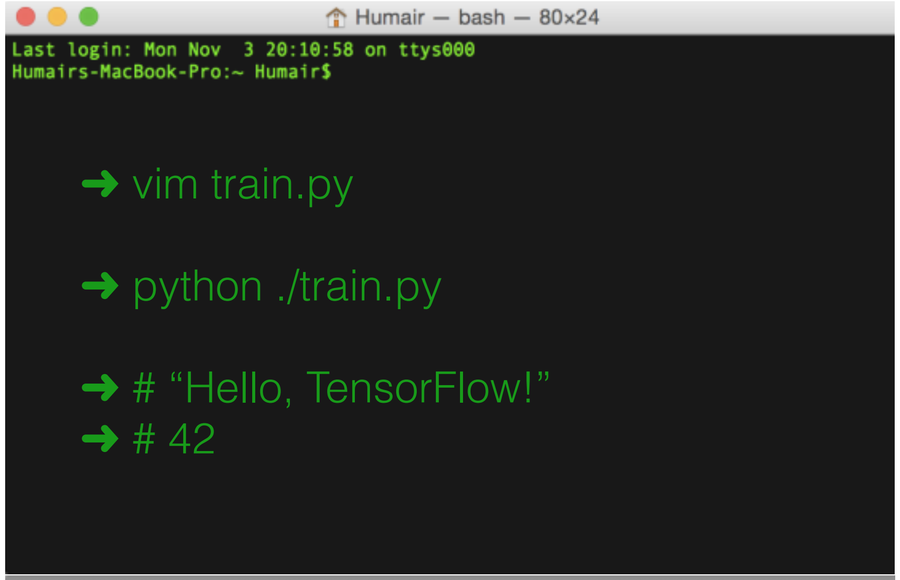
Introduce TensorFlow

```python
# Import the library
import tensorflow as tf

# Define the graph
hello_op = tf.constant('Hello, TensorFlow!')
a = tf.constant(10)
b = tf.constant(32)
compute_op = tf.add(a, b)

# Define the session to run graph
with tf.Session() as sess:
    print(sess.run(hello_op))
    print(sess.run(compute_op))
```



Introduce TensorFlow

```python
# Import the library
import tensorflow as tf

# Define the graph
hello_op = tf.constant('Hello, TensorFlow!')
a = tf.constant(10)
b = tf.constant(32)
compute_op = tf.add(a, b)

# Define the session to run graph
with tf.Session() as sess:
    print(sess.run(hello_op))
    print(sess.run(compute_op))
```



Summary of TensorFlow

TensorFlow

GIAC | BEIJING Dec.12.16-17

thegiac.com

Summary of TensorFlow

- TensorFlow is library

Summary of TensorFlow

- TensorFlow is library
- Installed manually

Summary of TensorFlow

- TensorFlow is library
- Installed manually
- Configure manually

Summary of TensorFlow

- TensorFlow is library
- Installed manually
- Configure manually
- Distributed manually

Summary of TensorFlow

- TensorFlow is library
- Installed manually
- Configure manually
- Distributed manually
- Run and tune manually

Standalone TensorFlow Workflow

Write the code

Ssh to server

Copy to server

Run and test

Update the code

Copy to server

Standalone TensorFlow Workflow

Write the code

Copy to server

Ssh to server

OOM

Run and test

Port used

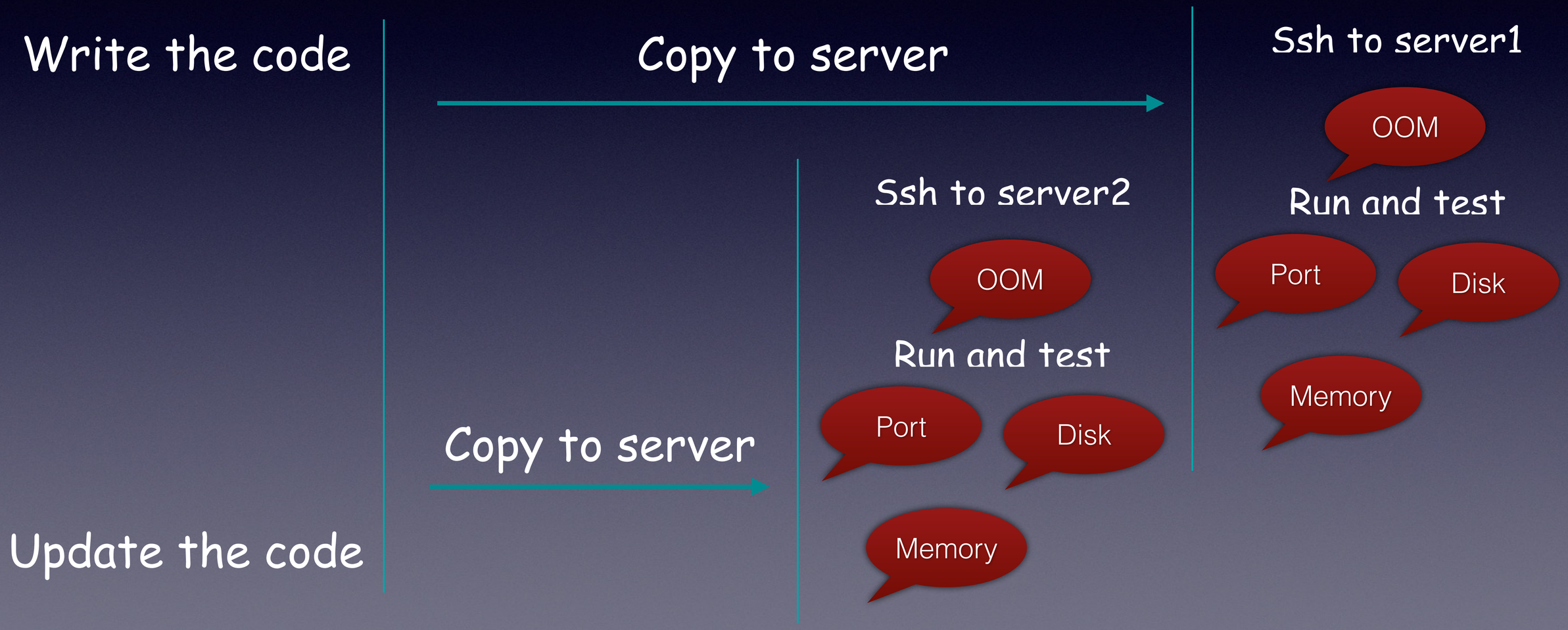
Disk fail

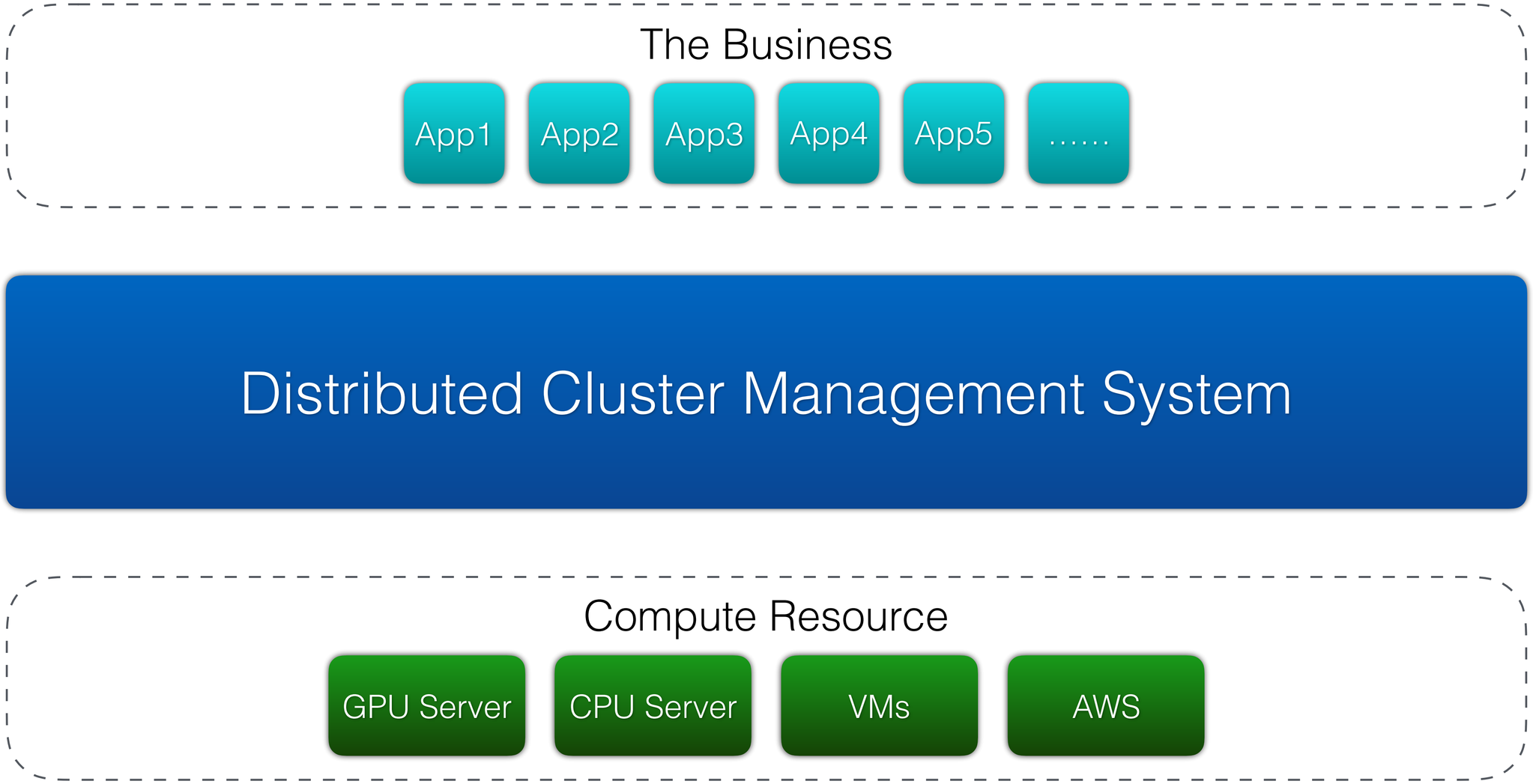
Memory leak

Update the code

Copy to server

Distributed TensorFlow Workflow




The Business
App1
App2
App3
App4
App5
……
Distributed Cluster Management System
Compute Resource
GPU Server
CPU Server
VMs
AWS

Agenda

- 机器学习与深度学习应用
- **深度学习平台架构与设计**
- 深度学习平台应用与实践

Cloud-ml: The Principles

Cloud-ml: The Principles

- Cloud computing instead of bare metal

Cloud-ml: The Principles

- Cloud computing instead of bare metal
- Submit the jobs instead of ssh + script

Cloud-ml: The Principles

- Cloud computing instead of bare metal
- Submit the jobs instead of ssh + script
- Integrated with train and model service

Cloud-ml: The Principles

- Cloud computing instead of bare metal
- Submit the jobs instead of ssh + script
- Integrated with train and model service
- High availability and failover for train job

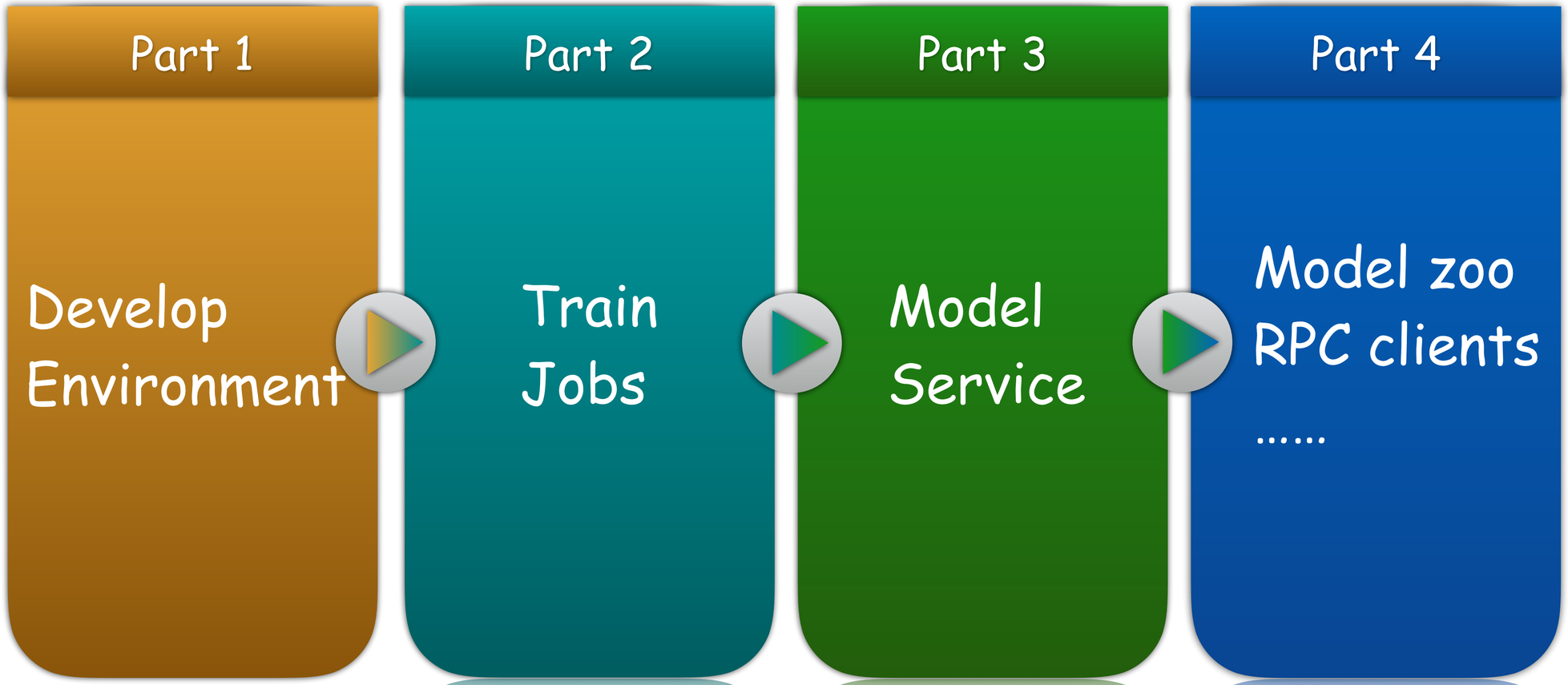
Cloud-ml: The Principles

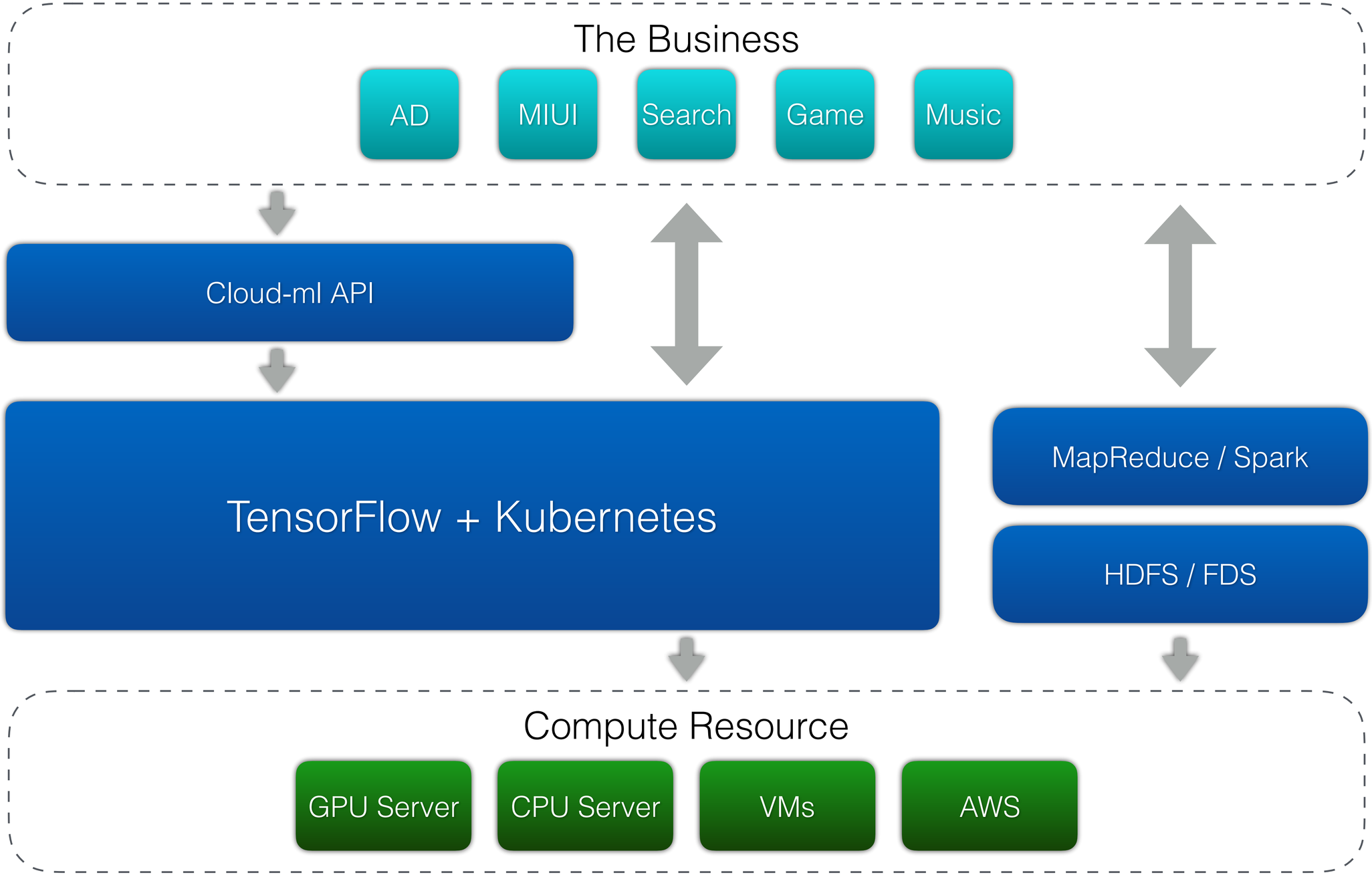
- Cloud computing instead of bare metal
- Submit the jobs instead of ssh + script
- Integrated with train and model service
- High availability and failover for train job
- Resource isolation and dynamic scheduling

Cloud-ml: The Principles

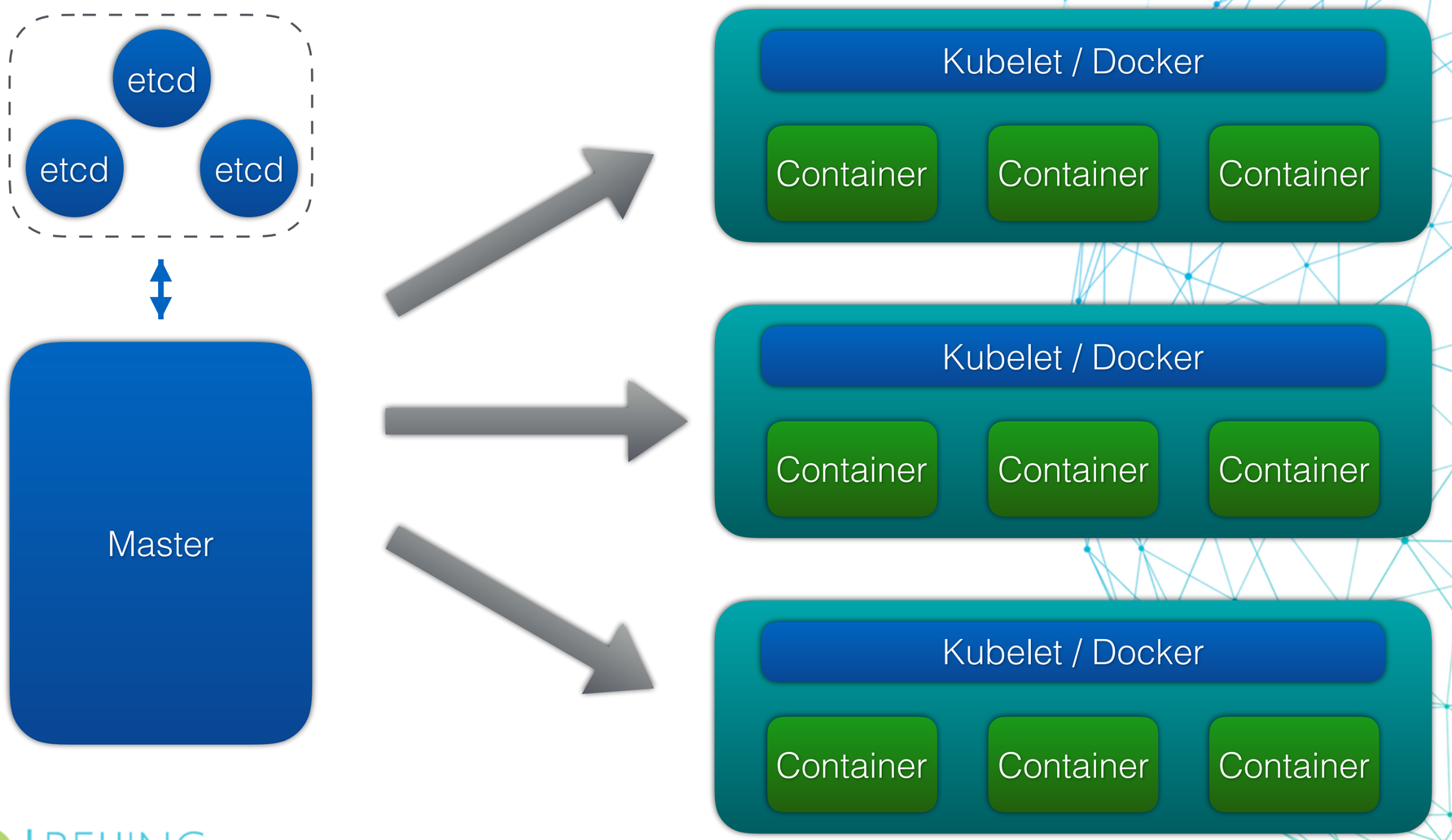
- Cloud computing instead of bare metal
- Submit the jobs instead of ssh + script
- Integrated with train and model service
- High availability and failover for train job
- Resource isolation and dynamic scheduling
- Parallel training and automatically tuning

Cloud-ml: All-in-one Platform

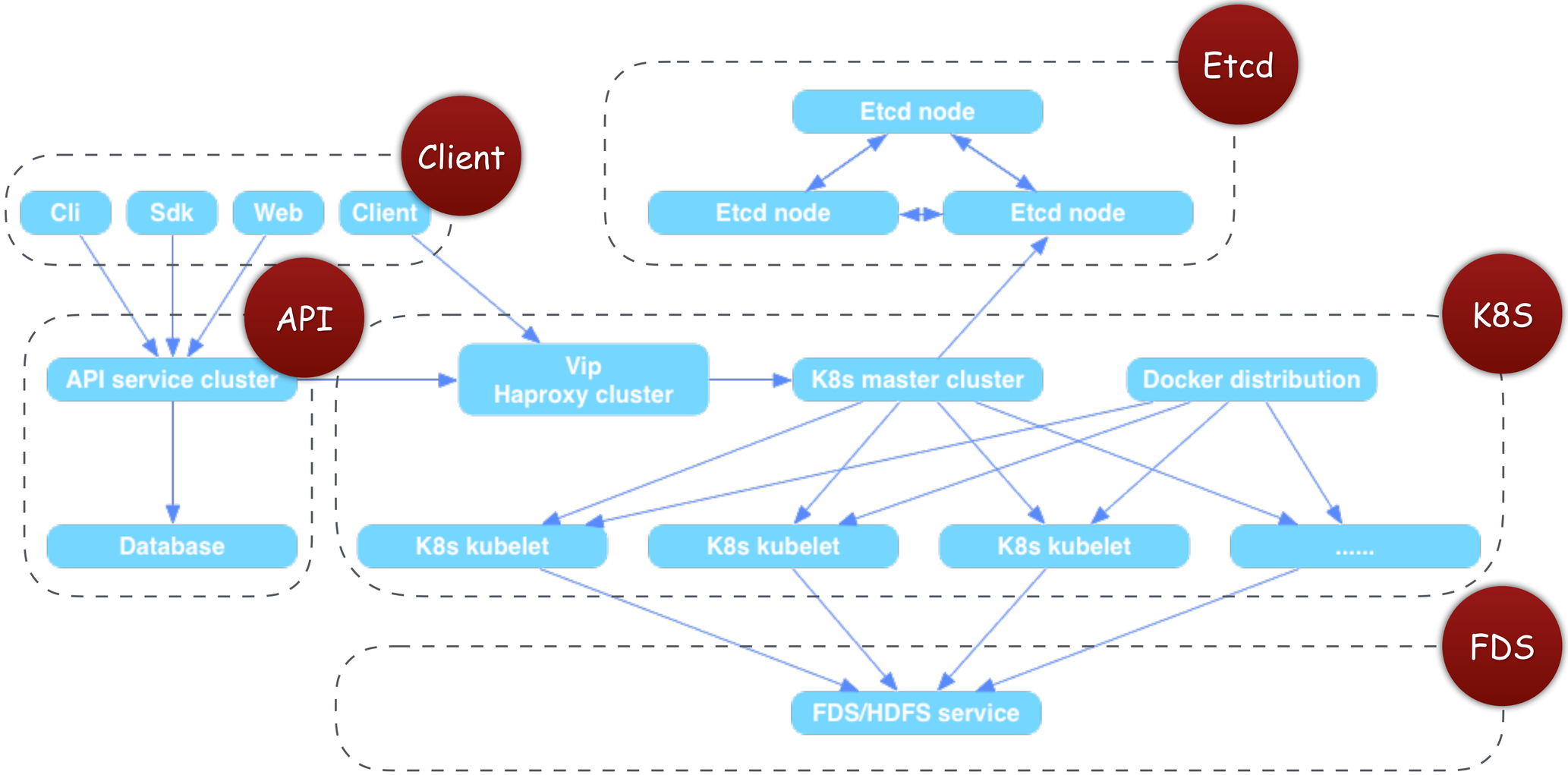



The Business
AD
MIUI
Search
Game
Music
Cloud-ml API
TensorFlow + Kubernetes
MapReduce / Spark
HDFS / FDS
Compute Resource
GPU Server
CPU Server
VMs
AWS

Cloud-ml: Kubernetes Inside



Cloud-ml: The Architecture



Cloud-ml: Submit Train Job

```
# Import the library
import tensorflow as tf

# Define the graph
hello_op = tf.constant('Hello, TensorFlow!')
a = tf.constant(10)
b = tf.constant(32)
compute_op = tf.add(a, b)

# Define the session to run graph
with tf.Session() as sess:
    print(sess.run(hello_op))
    print(sess.run(compute_op))
```

Cloud-ml: Train Job

API

```
POST /cloud_ml/v1/train

{
  "job_name": "face-recognition",
  "module_name": "trainer.task",
  "trainer_uri": "fds://cloudml/train
  "job_args": [],
  "output_path": "fds://cloudml/face-
  "master_spec": {
    "replica_count": 1
  }
}
```

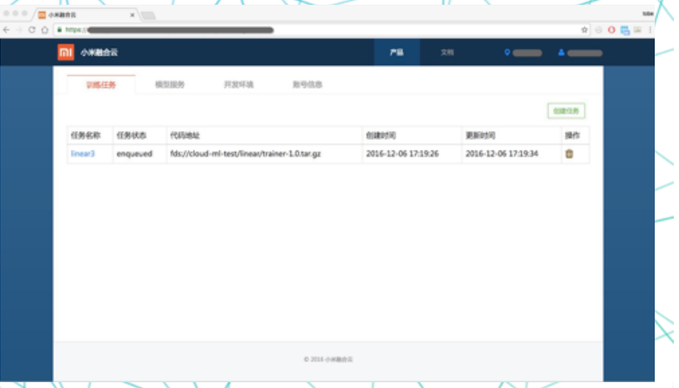
SDK

```
from cloud_ml.client import CloudMlClient
from cloud_ml.models.train_job import TrainJob

train_job = TrainJob('face-recognition', 'trai
            'fds://cloudml/trainer-1.0.tar.gz'
train_job.job_args = []
train_job.output_path = "fds://cloudml/face-mo
train_job.master_spec = {"replica_count": 1}

client = CloudMlClient('access_key', 'secret_k
client.submit_train_job(train_job)
```

Command

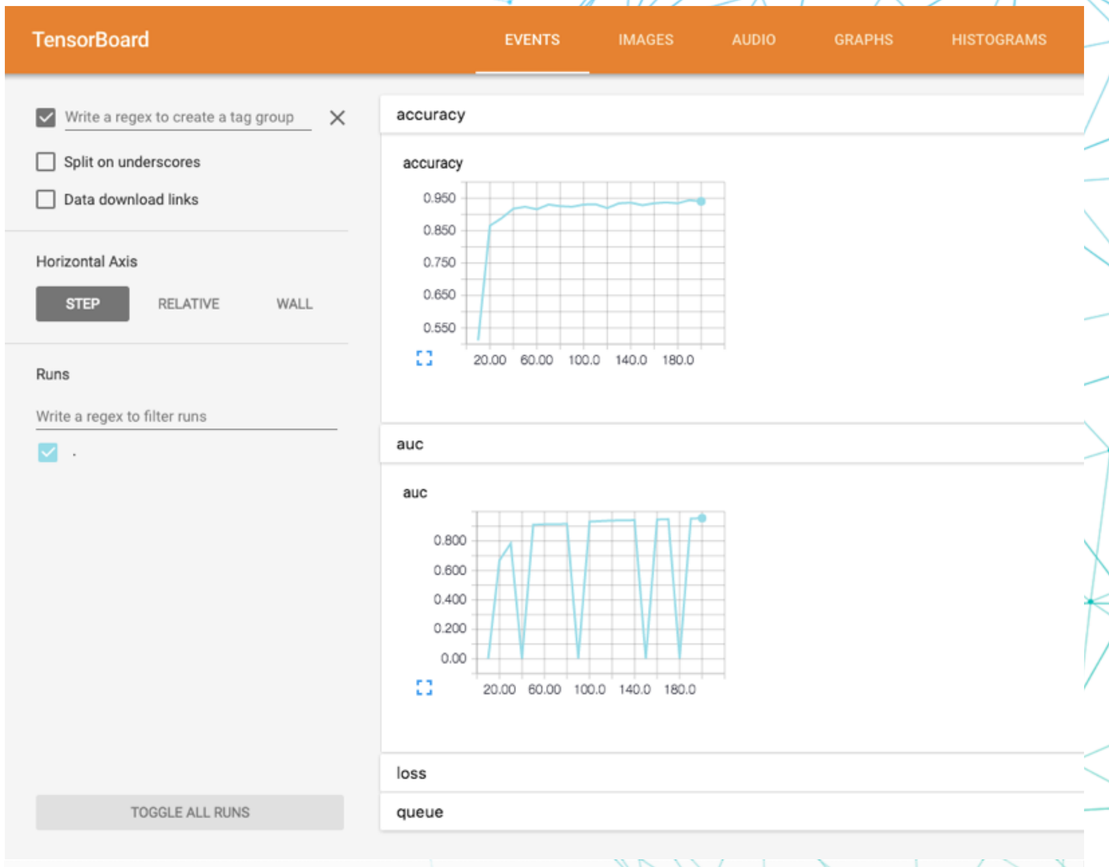
```
→ cloudml jobs submit "face-recognition"
    "trainer.task" "fds://cloudml/trainer

→ cloudml jobs cancel "face-recognition"

→ cloudml jobs list

→ cloudml models deploy "face-recognition
    "v1" "fds://cloudml/face-model"

→ cloudml models destroy "face-recognitio

→ cloudml models list
```

Web

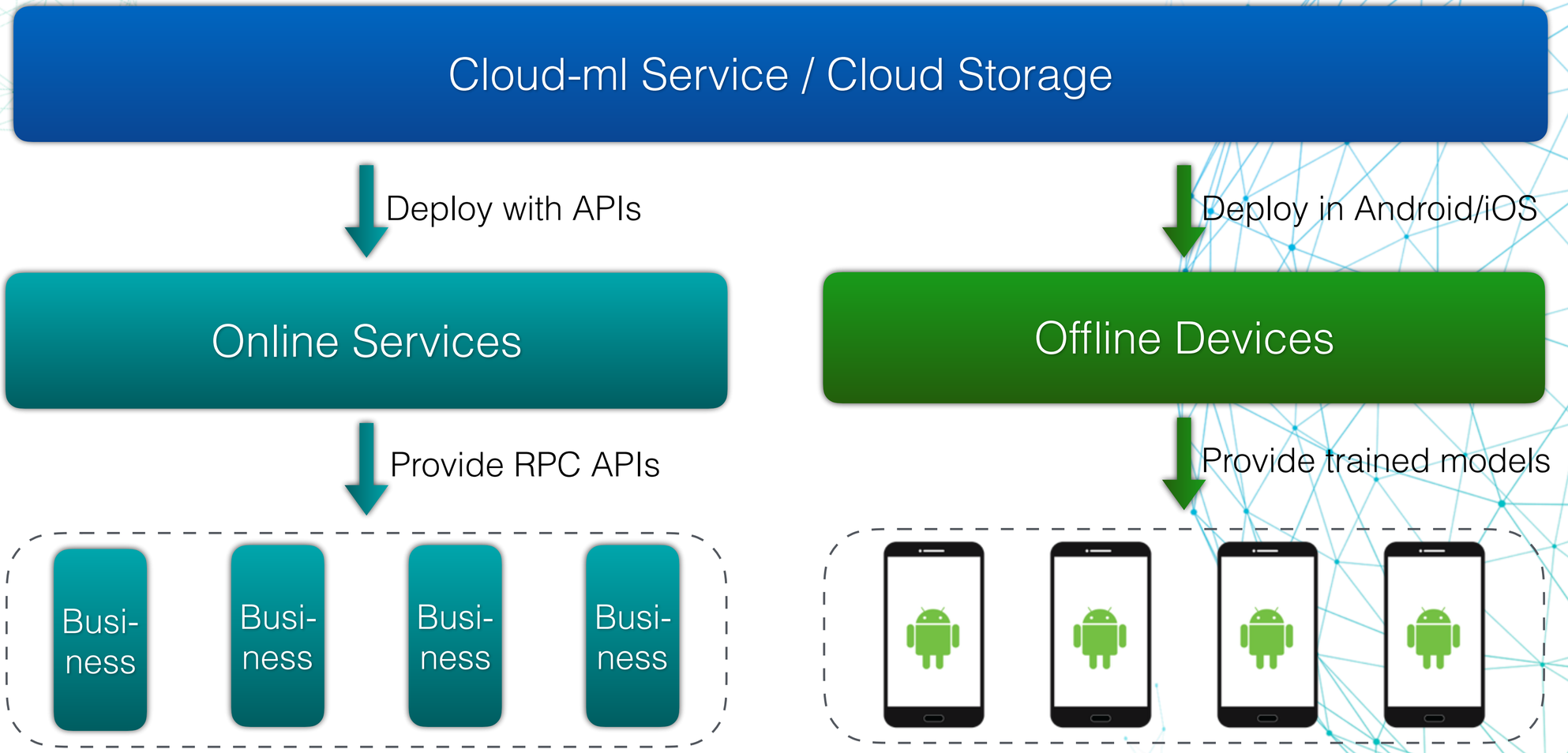


Cloud-ml: Train Job

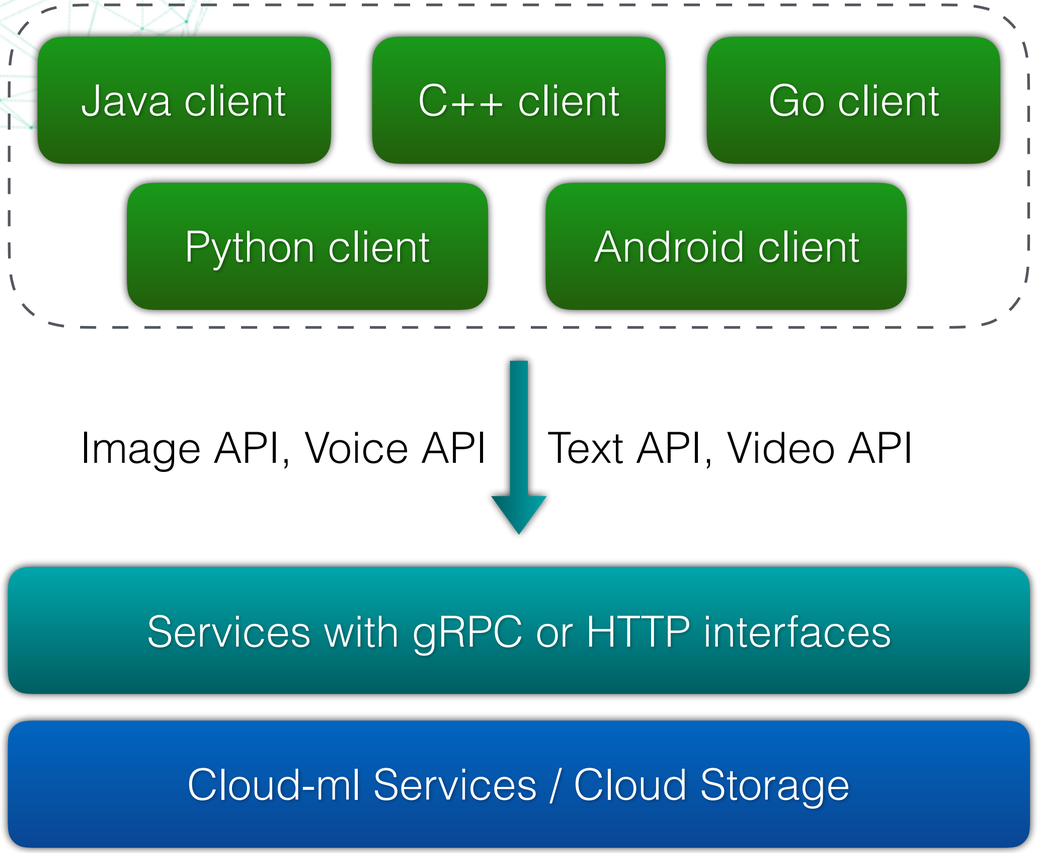
```
2016-10-26T02:56:47.561916526Z Processing trainer-1.0-py2.7.egg
2016-10-26T02:56:47.562317017Z Copying trainer-1.0-py2.7.egg to /usr/local/lib/pythor
2016-10-26T02:56:47.563445769Z Adding trainer 1.0 to easy-install.pth file
2016-10-26T02:56:47.564566658Z
2016-10-26T02:56:47.564574540Z Installed /usr/local/lib/python2.7/dist-packages/trair
2016-10-26T02:56:47.565210430Z Processing dependencies for trainer==1.0
2016-10-26T02:56:47.565416203Z Finished processing dependencies for trainer==1.0
2016-10-26T02:56:47.574070073Z INFO:root:Try to run python module: trainer.task
2016-10-26T02:56:53.037264387Z INFO:tensorflow:/tmp/linear_model/00000001-tmp/export-
2016-10-26T02:56:53.037331410Z INFO:tensorflow:/tmp/linear_model/00000001-tmp/export-
2016-10-26T02:56:53.259093203Z Use the optimizer: sgd
2016-10-26T02:56:53.259130161Z Save tensorboard files into: ./tensorboard/
2016-10-26T02:56:53.259157411Z Run training with epoch number: 10
2016-10-26T02:56:53.259163786Z Epoch: 0, loss: 5.55905914307
2016-10-26T02:56:53.259179744Z Epoch: 1, loss: 3.98923826218
2016-10-26T02:56:53.259185265Z Epoch: 2, loss: 1.15070474148
2016-10-26T02:56:53.259190556Z Epoch: 3, loss: 0.256429493427
2016-10-26T02:56:53.259195798Z Epoch: 4, loss: 0.0424121692777
2016-10-26T02:56:53.259201130Z Epoch: 5, loss: 0.00265768845566
2016-10-26T02:56:53.259206653Z Epoch: 6, loss: 0.000737804220989
2016-10-26T02:56:53.259211961Z Epoch: 7, loss: 0.00451849261299
2016-10-26T02:56:53.259217125Z Epoch: 8, loss: 0.0076722134836
2016-10-26T02:56:53.259222407Z Epoch: 9, loss: 0.00959475897253
2016-10-26T02:56:53.259227708Z [0:00:02.928687] End of standalone training.
2016-10-26T02:56:53.259235015Z Get the model, w: 1.88596236706, b: 9.95958137512
2016-10-26T02:56:53.259240345Z Exporting trained model to /tmp/linear_model/
2016-10-26T02:56:53.259246348Z Done exporting!
```



Cloud-ml: Model Service

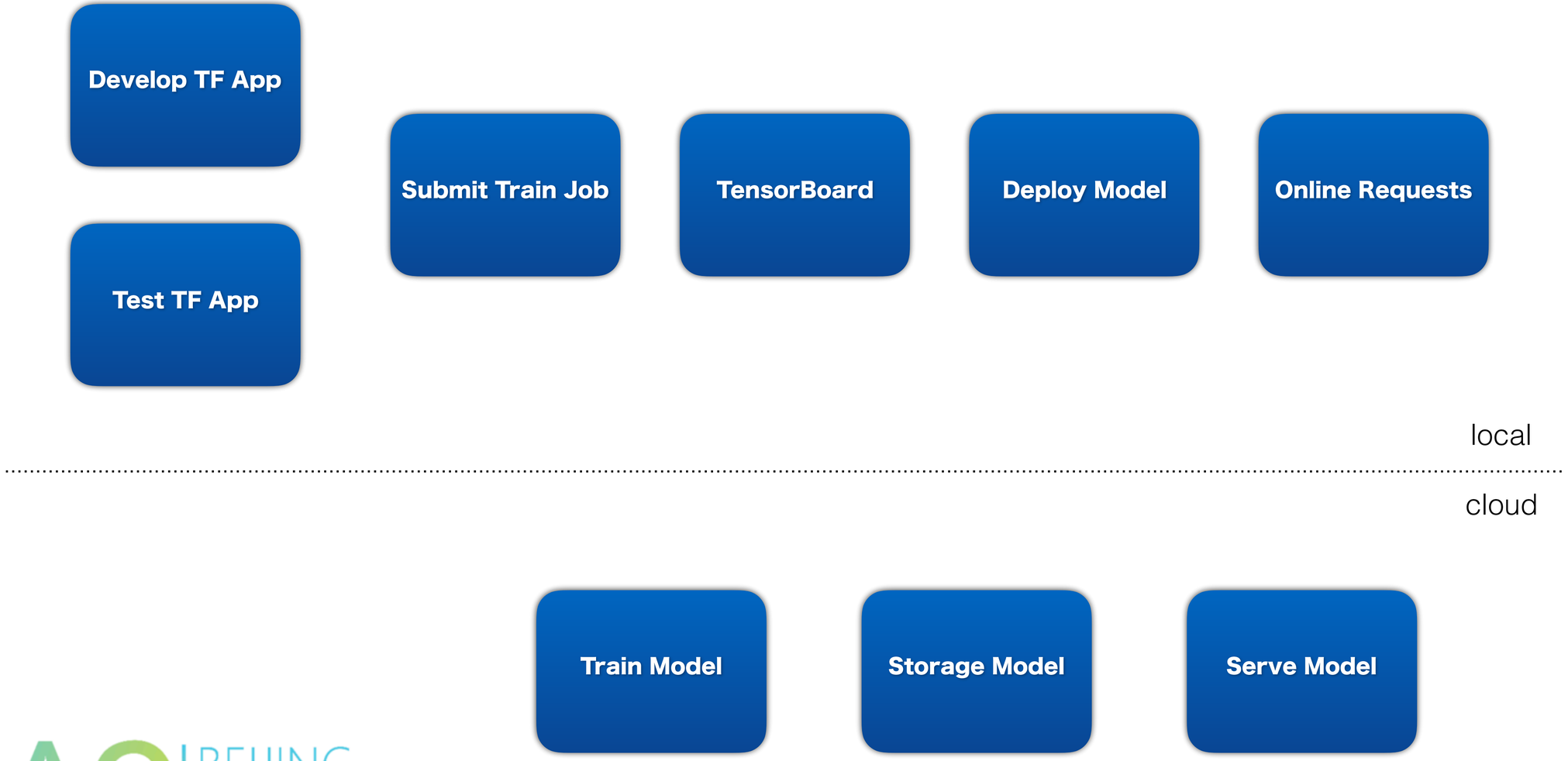


Cloud-ml: Predict Client

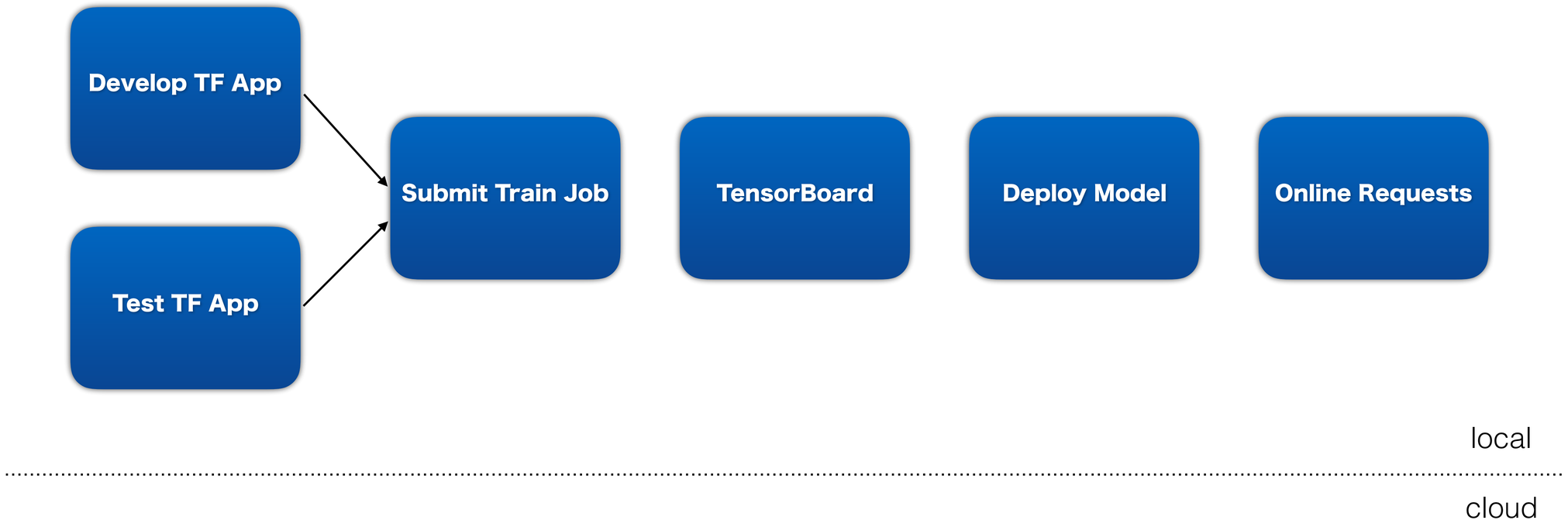



```python
def main():
    # Connect with the gRPC server
    server_address = "127.0.0.1:50051"
    request_timeout = 5.0
    channel = grpc.insecure_channel(server_address)
    stub = predict_pb2.PredictionServiceStub(channel)

    # Make request data
    request = predict_pb2.PredictRequest()
    samples_features = np.array(
        [[10, 10, 10, 8, 6, 1, 8, 9, 1], [10, 10, 10, 8, 6, 1, 8, 9, 1]])
    # Convert numpy to TensorProto
    request.inputs["features"].CopyFrom(tensor_util.make_tensor_proto(
        samples_features))

    # Invoke gRPC request
    response = stub.Predict(request, request_timeout)

    # Convert TensorProto to numpy
    result = {}
    for k, v in response.outputs.items():
        result[k] = tensor_util.MakeNdarray(v)
    print(result)
```

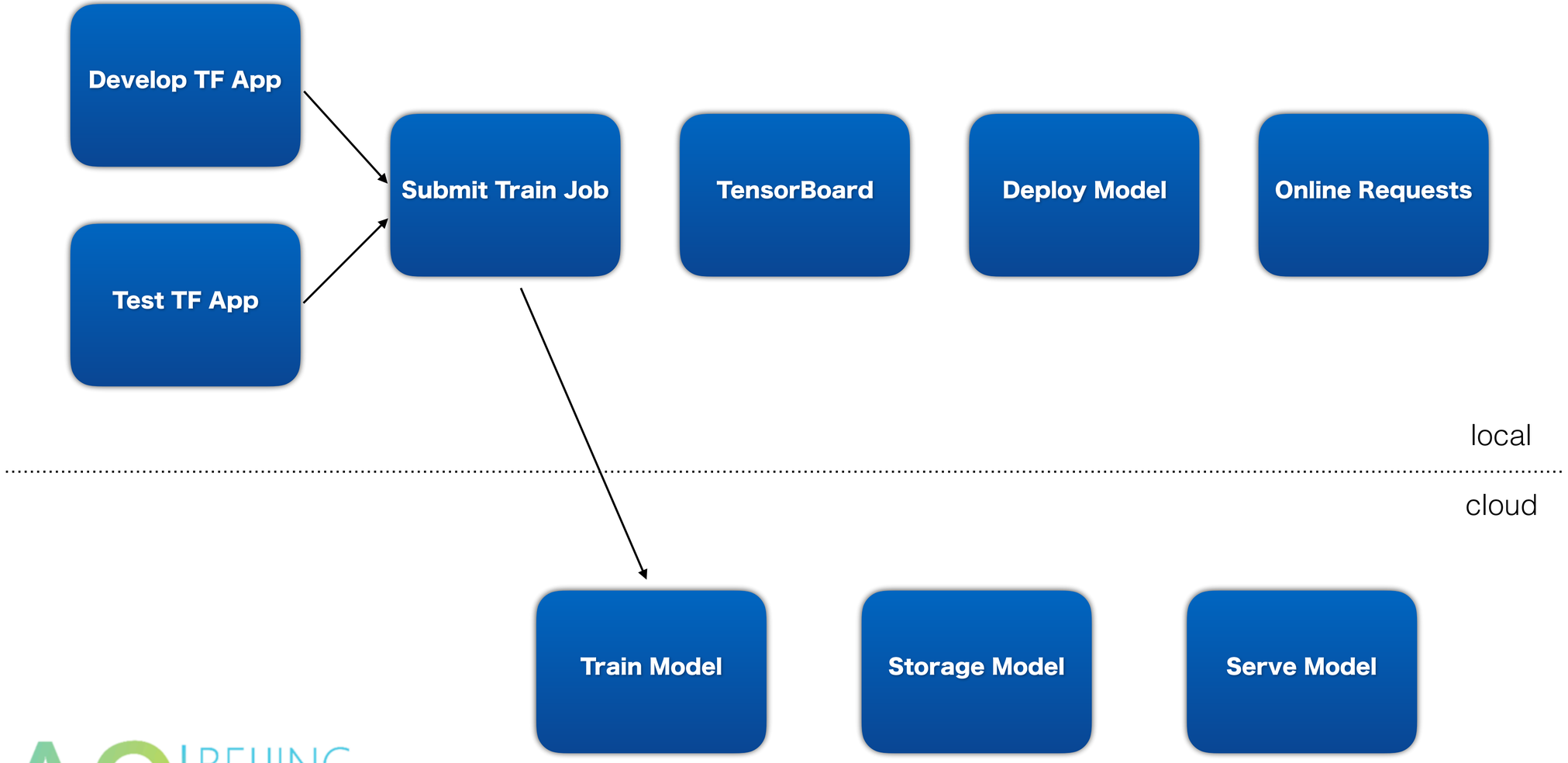
Cloud-ml: Wrap-up



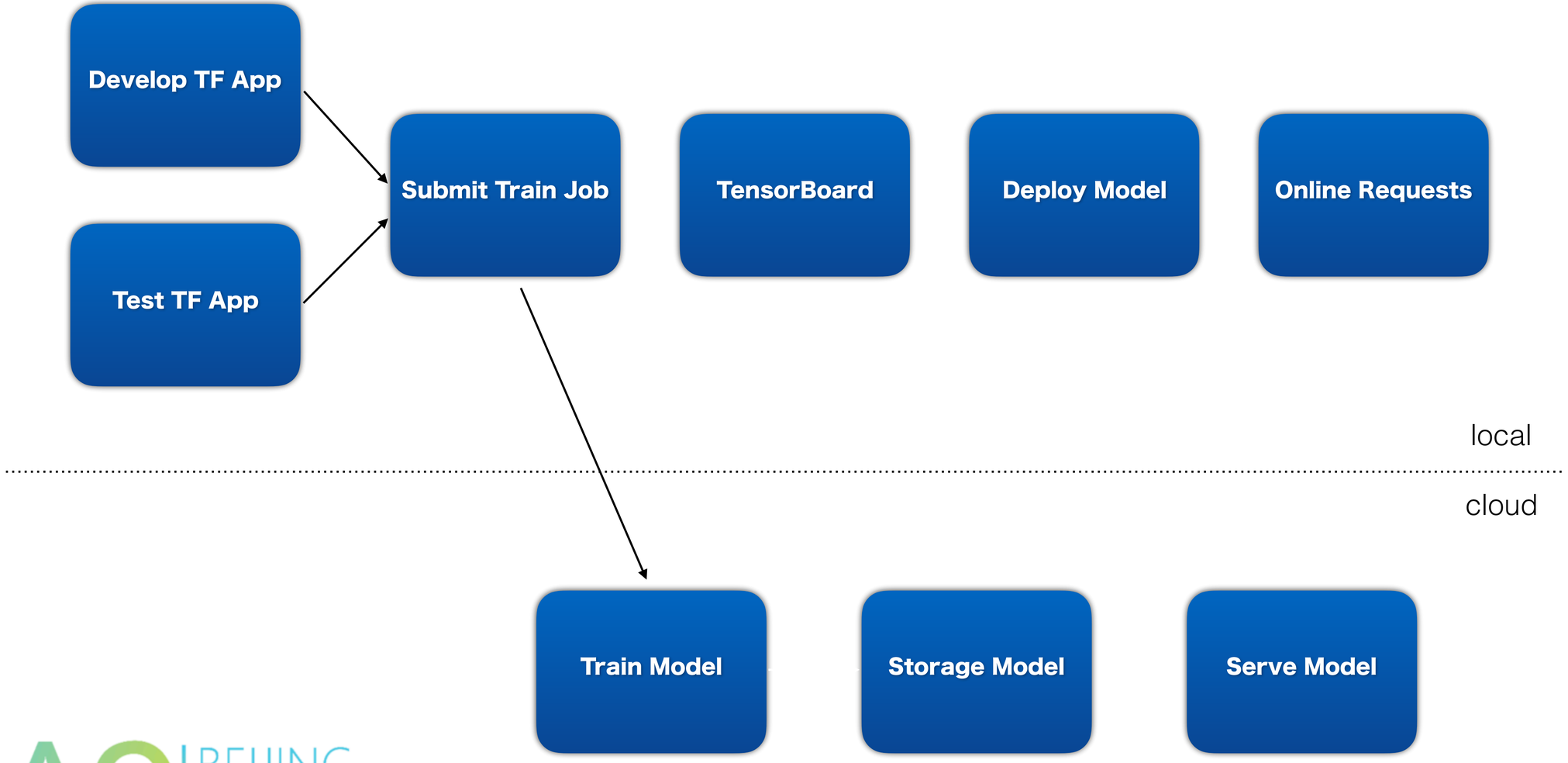
Cloud-ml: Wrap-up



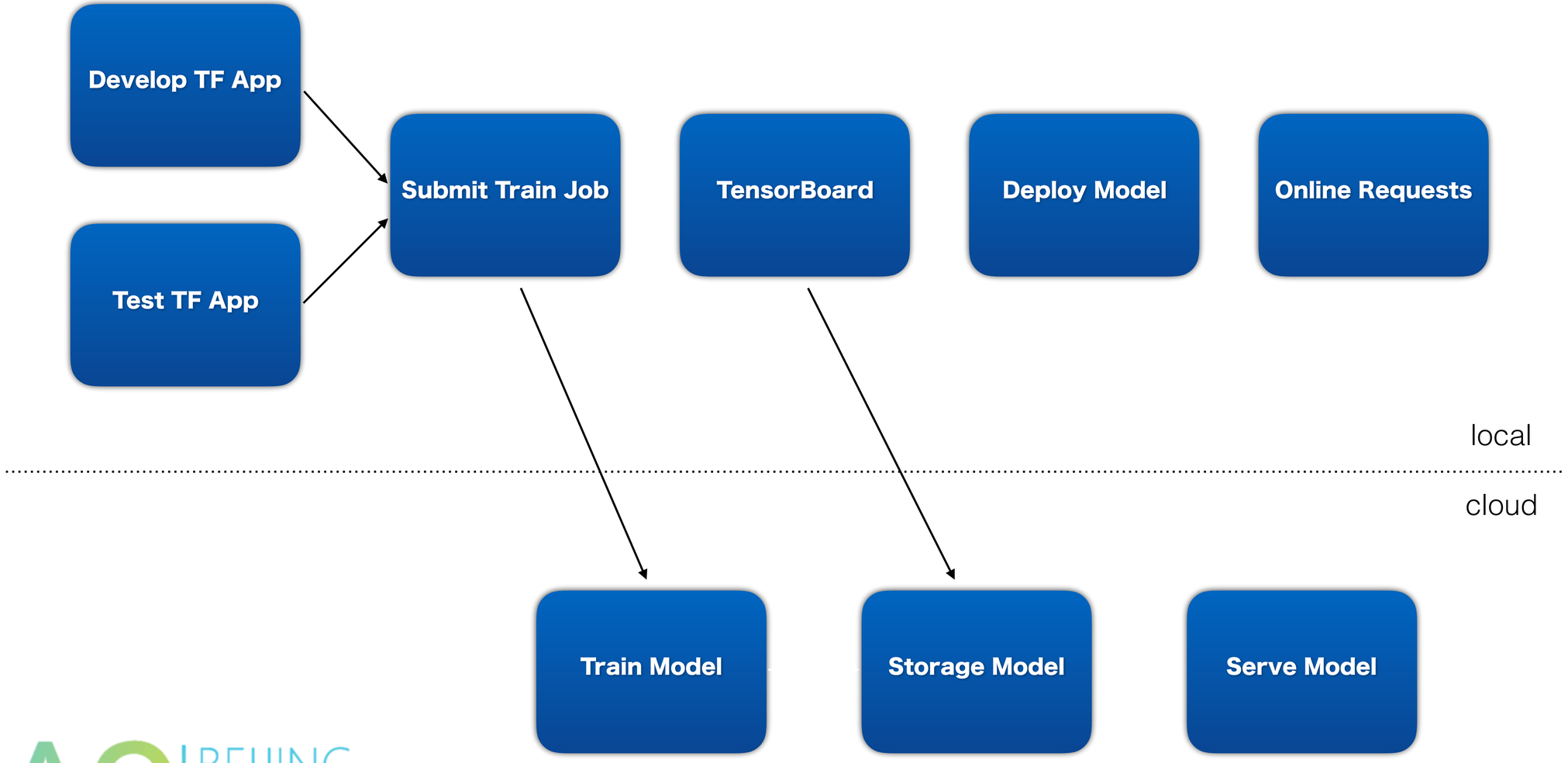
Cloud-ml: Wrap-up



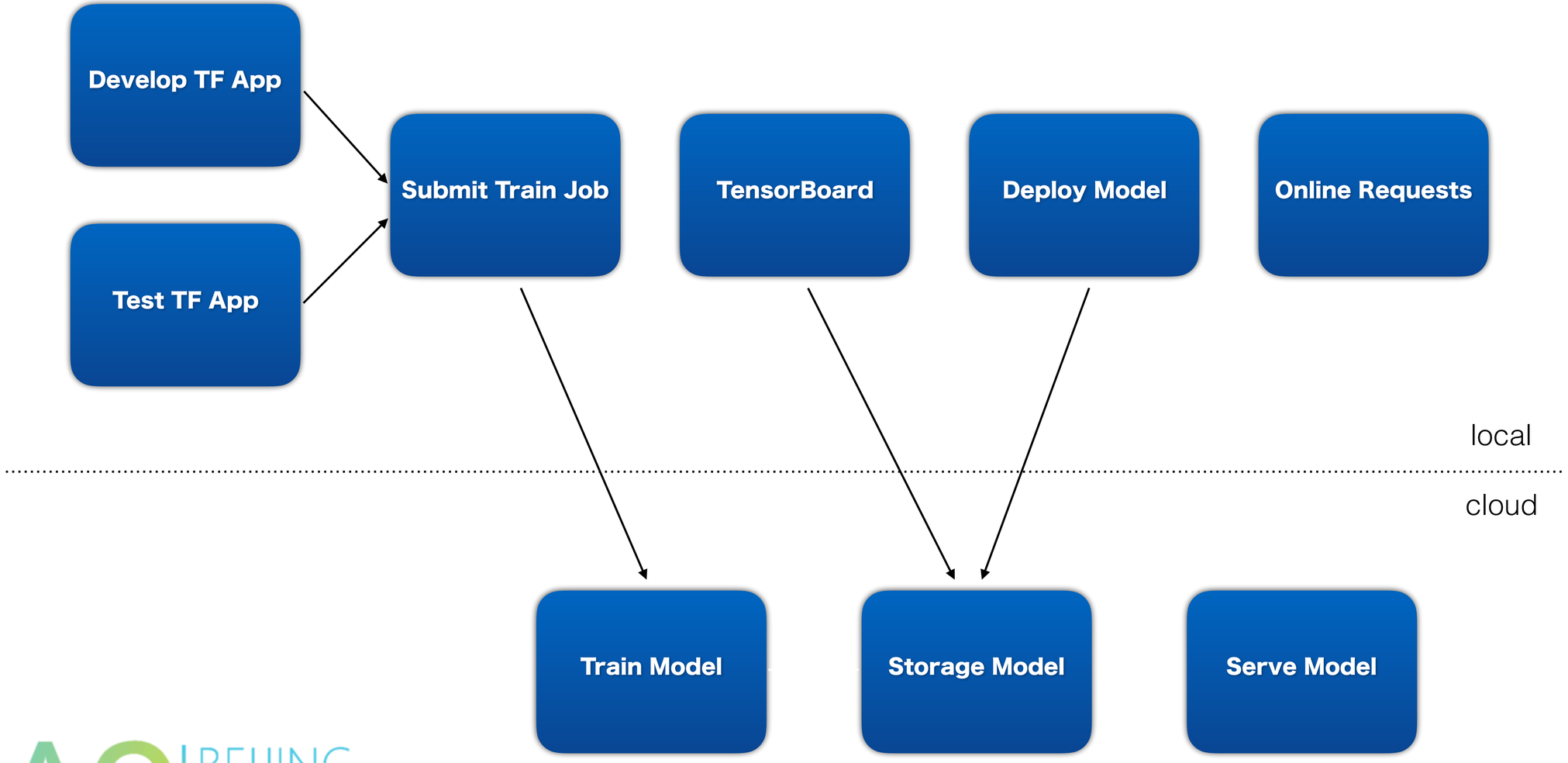
Cloud-ml: Wrap-up



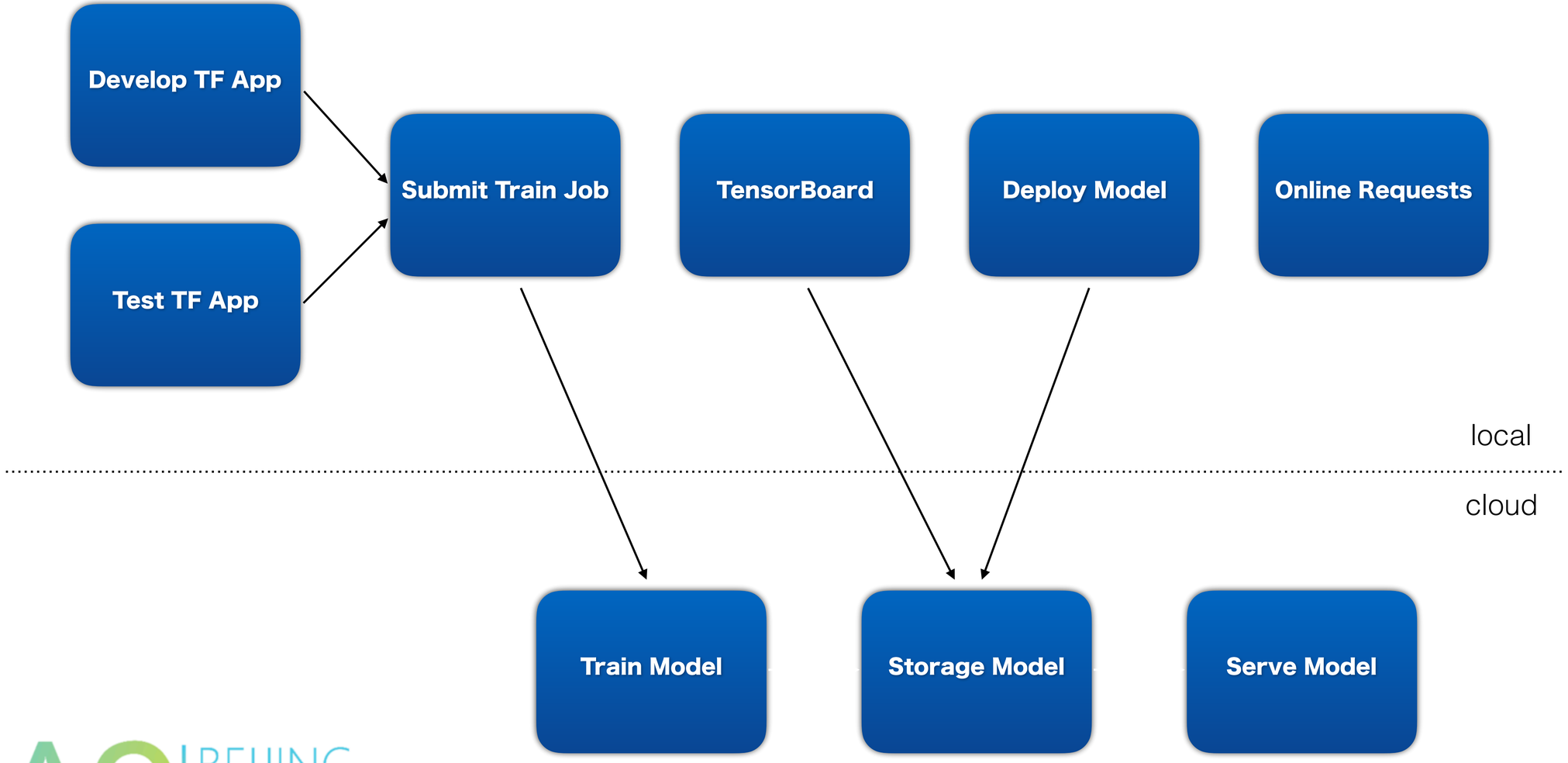
Cloud-ml: Wrap-up



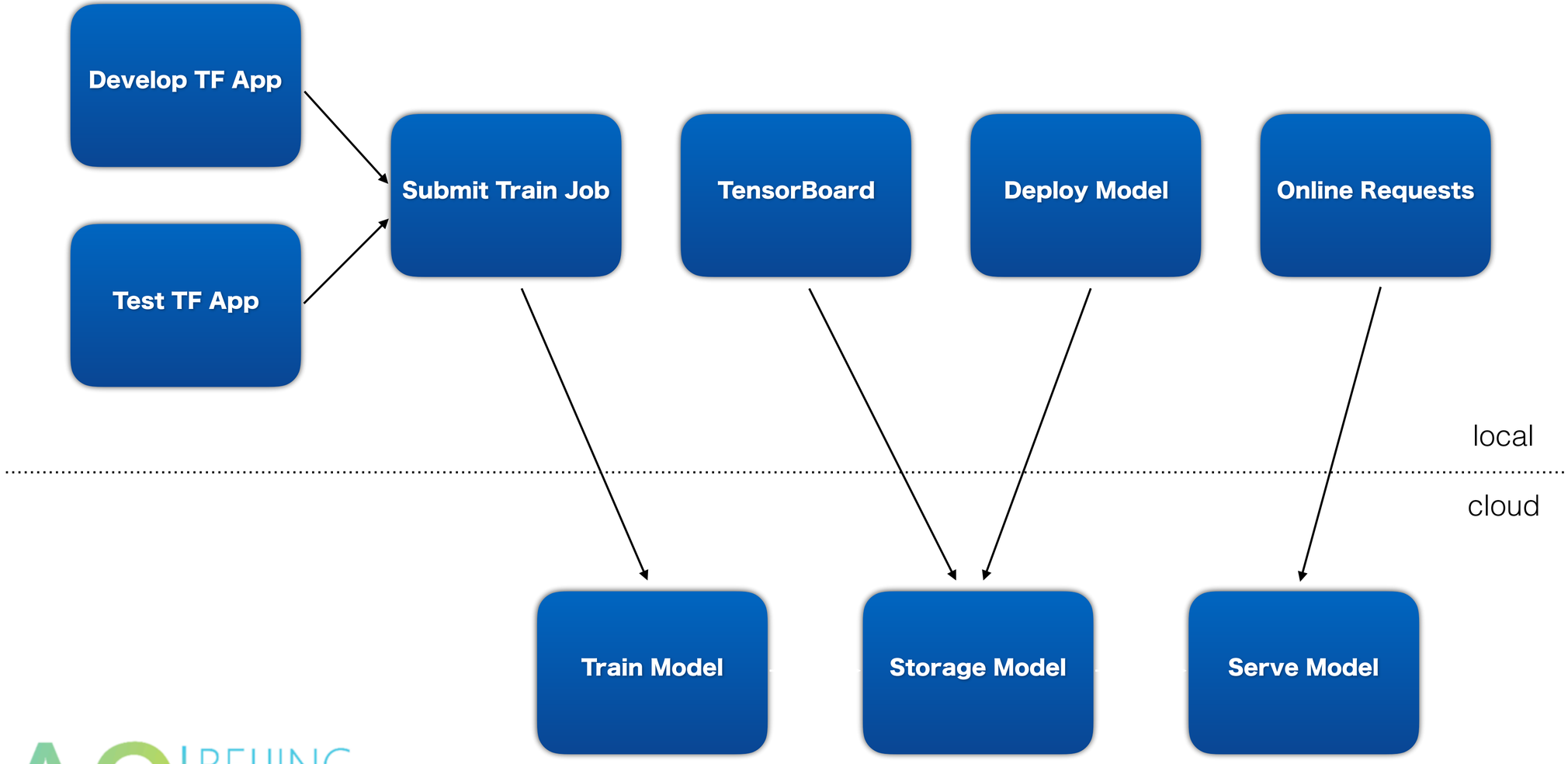
Cloud-ml: Wrap-up



Cloud-ml: Wrap-up



Cloud-ml: Wrap-up



Agenda

- 机器学习与深度学习应用
- 深度学习平台架构与设计
- **深度学习平台应用与实践**

Practice: Distributed Training

```
python -m trainer.task

--ps_hosts=127.0.0.1:30001

--worker_hosts=127.0.0.1:3002

--master_hosts=127.0.0.1:3003

--job_name=master

--task_index=0
```

```
TF_CONFIG=
'{
    "cluster": {
        "ps": ["127.0.0.1:3001"],
        "worker": ["127.0.0.1:3002"],
        "master": ["127.0.0.1:3003"]
    },
    "task": {
        "index": 0,
        "type": "master"
    }
}'

python -m trainer.task
```

https://github.com/tobegit3hub/deep_recommend_system/

https://github.com/tobegit3hub/distributed_tensorflow

Practice: Distributed Training

```
TF_CONFIG=
'{
    "cluster": {
        "ps": ["127.0.0.1:3001"],
        "worker": ["127.0.0.1:3002"],
        "master": ["127.0.0.1:3003"]
    },
    "task": {
        "index": 0,
        "type": "master"
    }
}'

python -m trainer.task
```

2 PS
2 Worker
1 Master

cloud-ml

```
{
  "ps": ["127.0.0.1:3001", "127.0.0.1:3002"],
  "worker": ["127.0.0.1:3003", "127.0.0.1:3004"],
  "master": ["127.0.0.1:3005"]
}
```

Train Job(PS 0)
Train Job(PS 1)
Train Job(Worker 0)
Train Job(Worker 1)
Train Job(Master 0)

Practice: Storage Integration

Practice: Storage Integration

- cloudml jobs submit -n linear -m trainer.task -u fds://cloud-ml-test/linear/trainer-1.0.tar.gz -c 0.5 -M 100M -g 1

Practice: Storage Integration

- cloudml jobs submit -n linear -m trainer.task -u fds://cloud-ml-test/linear/trainer-1.0.tar.gz -c 0.5 -M 100M -g 1
- tensorboard --logdir fds://cloud-ml-test/linear/tensorboard/

Practice: Storage Integration

- cloudml jobs submit -n linear -m trainer.task -u fds://cloud-ml-test/linear/trainer-1.0.tar.gz -c 0.5 -M 100M -g 1
- tensorboard --logdir fds://cloud-ml-test/linear/tensorboard/
- cloudml models create -n linear_model -v v1 -u fds://cloud-ml-test/linear_model

Practice: Storage Integration

- cloudml jobs submit -n linear -m trainer.task -u fds://cloud-ml-test/linear/trainer-1.0.tar.gz -c 0.5 -M 100M -g 1
- tensorboard --logdir fds://cloud-ml-test/linear/tensorboard/
- cloudml models create -n linear_model -v v1 -u fds://cloud-ml-test/linear_model
- cloudml models predict linear_model v1 -d ./data.json

Practice: Storage Integration

- cloudml jobs submit -n linear -m trainer.task -u fds://cloud-ml-test/linear/trainer-1.0.tar.gz -c 0.5 -M 100M -g 1
- tensorboard --logdir fds://cloud-ml-test/linear/tensorboard/
- cloudml models create -n linear_model -v v1 -u fds://cloud-ml-test/linear_model
- cloudml models predict linear_model v1 -d ./data.json

```
{
    "keys_dtype": "int32",
    "keys": [[1], [2]],
    "X_dtype": "float32",
    "X": [[10.0], [30.0]]
}
```

Practice: Multi-tenantcy

- Authentication and Authorization
- Delegate credential for users
- Resource isolation with container
- Resource quota for tenants

	Memory / Used	CPU / Used	GPU / Used
Train job	10G 0M 0.0K / 300M 0.0K	10.0 / 0.6	0 / 0
Model service	10G 0M 0.0K / 0.0K	10.0 / 0.0	0 / 0
Dev environment	10G 0M 0.0K / 0.0K	10.0 / 0.0	0 / 0

Practice: GPU Support

May be released in Kubernetes 1.6

Pull request from kubernetes/kubernetes#28216

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: nvidia-gpu-test
spec:
  containers:
  - name: nvidia-gpu
    image: tensorflow/tensorflow:latest-gpu
    resources:
      limits:
        alpha.kubernetes.io/nvidia-gpu: 1
```

Practice: GPU Support

May be released in Kubernetes 1.6

Pull request from kubernetes/kubernetes#28216

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: nvidia-gpu-test
spec:
  containers:
  - name: nvidia-gpu
    image: tensorflow/tensorflow:latest-gpu
    resources:
      limits:
        alpha.kubernetes.io/nvidia-gpu: 1
```

Practice: Support HPAT

- HyperParameters Automatically Tuning
- Training and tuning models concurrently

```
## Submit job for hyparameters automatically tunning

{
  "job_name": "linear37",
  "module_name": "trainer.task",
  "trainer_uri": "fds://cloud-ml-test/linear/trainer-1.0.tar.gz",
  "job_args": "--max_epochs 5000",
  "cpu_limit": "0.5",
  "memory_limit": "100M",
  "hyperparameters": {
    "goal": "MINIMIZE",
    "output_path": "fds://cloud-ml-test/linear/linear_output37",
    "params": [
      {"optimizer": "ftrl", "learning_rate": 0.1},
      {"optimizer": "ftrl", "learning_rate": 0.5},
      {"optimizer": "sgd", "learning_rate": 0.1},
      {"optimizer": "sgd", "learning_rate": 0.5}
    ]
  }
}
```

```
➜  command git:(master) ✗ ./command.py jobs hp linear37-hp-3
INFO:requests.packages.urllib3.connectionpool:Starting new HTTP connection (1): 127.0.0.1
Goal: MINIMIZE
Trial count: 4
Best trial:
        Metrics: 0.0888650268316
        Params: --output_path=fds://cloud-ml-test/linear/linear_output37/0 --learning_rate=0.1 --optimizer=ftrl
        Step: 20000
Trial 0:
        Metrics: 0.0888650268316
        Params: --output_path=fds://cloud-ml-test/linear/linear_output37/0 --learning_rate=0.1 --optimizer=ftrl
        Step: 20000
Trial 1:
        Metrics: 0.0888650268316
        Params: --output_path=fds://cloud-ml-test/linear/linear_output37/1 --learning_rate=0.5 --optimizer=ftrl
        Step: 20000
Trial 2:
        Metrics: 0.0888650268316
        Params: --output_path=fds://cloud-ml-test/linear/linear_output37/2 --learning_rate=0.1 --optimizer=sgd
        Step: 20000
Trial 3:
        Metrics: 0.0888650268316
        Params: --output_path=fds://cloud-ml-test/linear/linear_output37/3 --learning_rate=0.5 --optimizer=sgd
        Step: 20000
```

Practice: Support HPAT

- HyperParameters Automatically Tuning
- Training and tuning models concurrently

```
## Submit job for hyparameters automatically tunning

{
  "job_name": "linear37",
  "module_name": "trainer.task",
  "trainer_uri": "fds://cloud-ml-test/linear/trainer-1.0.tar.gz",
  "job_args": "--max_epochs 5000",
  "cpu_limit": "0.5",
  "memory_limit": "100M",
  "hyperparameters": {
    "goal": "MINIMIZE",
    "output_path": "fds://cloud-ml-test/linear/linear_output37",
    "params": [
      {"optimizer": "ftrl", "learning_rate": 0.1},
      {"optimizer": "ftrl", "learning_rate": 0.5},
      {"optimizer": "sgd", "learning_rate": 0.1},
      {"optimizer": "sgd", "learning_rate": 0.5}
    ]
  }
}
```

```
➜  command git:(master) ✗ ./command.py jobs hp linear37-hp-3
INFO:requests.packages.urllib3.connectionpool:Starting new HTTP connection (1): 127.0.0.1
Goal: MINIMIZE
Trial count: 4
Best trial:
        Metrics: 0.0888650268316
        Params: --output_path=fds://cloud-ml-test/linear/linear_output37/0 --learning_rate=0.1 --optimizer=ftrl
        Step: 20000
Trial 0:
        Metrics: 0.0888650268316
        Params: --output_path=fds://cloud-ml-test/linear/linear_output37/0 --learning_rate=0.1 --optimizer=ftrl
        Step: 20000
Trial 1:
        Metrics: 0.0888650268316
        Params: --output_path=fds://cloud-ml-test/linear/linear_output37/1 --learning_rate=0.5 --optimizer=ftrl
        Step: 20000
Trial 2:
        Metrics: 0.0888650268316
        Params: --output_path=fds://cloud-ml-test/linear/linear_output37/2 --learning_rate=0.1 --optimizer=sgd
        Step: 20000
Trial 3:
        Metrics: 0.0888650268316
        Params: --output_path=fds://cloud-ml-test/linear/linear_output37/3 --learning_rate=0.5 --optimizer=sgd
        Step: 20000
```

Practice: Dependency management

- Dependency hell
- Environment isolation
- Standard Python package
- Installed before training

```
2016-10-26T02:56:47.561916526Z Processing trainer-1.0-py2.7.egg
2016-10-26T02:56:47.562317017Z Copying trainer-1.0-py2.7.egg to /usr/local/lib/pytho
2016-10-26T02:56:47.563445769Z Adding trainer 1.0 to easy-install.pth file
2016-10-26T02:56:47.564566658Z
2016-10-26T02:56:47.564574540Z Installed /usr/local/lib/python2.7/dist-packages/trai
2016-10-26T02:56:47.565210430Z Processing dependencies for trainer==1.0
2016-10-26T02:56:47.565416203Z Finished processing dependencies for trainer==1.0
2016-10-26T02:56:47.574070073Z INFO:root:Try to run python module: trainer.task
2016-10-26T02:56:53.037264387Z INFO:tensorflow:/tmp/linear_model/00000001-tmp/export-
2016-10-26T02:56:53.037331410Z INFO:tensorflow:/tmp/linear_model/00000001-tmp/export-
2016-10-26T02:56:53.259093203Z Use the optimizer: sgd
2016-10-26T02:56:53.259130161Z Save tensorboard files into: ./tensorboard/
2016-10-26T02:56:53.259157411Z Run training with epoch number: 10
2016-10-26T02:56:53.259163786Z Epoch: 0, loss: 5.55905914307
2016-10-26T02:56:53.259179744Z Epoch: 1, loss: 3.98923826218
2016-10-26T02:56:53.259185265Z Epoch: 2, loss: 1.15070474148
2016-10-26T02:56:53.259190556Z Epoch: 3, loss: 0.256429493427
2016-10-26T02:56:53.259195798Z Epoch: 4, loss: 0.0424121692777
2016-10-26T02:56:53.259201130Z Epoch: 5, loss: 0.00265768845566
2016-10-26T02:56:53.259206653Z Epoch: 6, loss: 0.000737804220989
2016-10-26T02:56:53.259211961Z Epoch: 7, loss: 0.00451849261299
2016-10-26T02:56:53.259217125Z Epoch: 8, loss: 0.0076722134836
2016-10-26T02:56:53.259222407Z Epoch: 9, loss: 0.00959475897253
2016-10-26T02:56:53.259227708Z [0:00:02.928687] End of standalone training.
2016-10-26T02:56:53.259235015Z Get the model, w: 1.88596236706, b: 9.95958137512
2016-10-26T02:56:53.259240345Z Exporting trained model to /tmp/linear_model/
2016-10-26T02:56:53.259246348Z Done exporting!
```

Practice: Dependency management

- Dependency hell
- Environment isolation
- Standard Python package
- Installed before training

```
2016-10-26T02:56:47.561916526Z Processing trainer-1.0-py2.7.egg
2016-10-26T02:56:47.562317017Z Copying trainer-1.0-py2.7.egg to /usr/local/lib/python
2016-10-26T02:56:47.563445769Z Adding trainer 1.0 to easy-install.pth file
2016-10-26T02:56:47.5645666587
2016-10-26T02:56:47.564574540Z Installed /usr/local/lib/python2.7/dist-packages/trai
2016-10-26T02:56:47.565210430Z Processing dependencies for trainer==1.0
2016-10-26T02:56:47.565416203Z Finished processing dependencies for trainer==1.0
2016-10-26T02:56:47.574070073Z INFO:root:Try to run python module: trainer.task
2016-10-26T02:56:53.037264387Z INFO:tensorflow:/tmp/linear_model/00000001-tmp/export-
2016-10-26T02:56:53.037331410Z INFO:tensorflow:/tmp/linear_model/00000001-tmp/export-
2016-10-26T02:56:53.259093203Z Use the optimizer: sgd
2016-10-26T02:56:53.259130161Z Save tensorboard files into: ./tensorboard/
2016-10-26T02:56:53.259157411Z Run training with epoch number: 10
2016-10-26T02:56:53.259163786Z Epoch: 0, loss: 5.55905914307
2016-10-26T02:56:53.259179744Z Epoch: 1, loss: 3.98923826218
2016-10-26T02:56:53.259185265Z Epoch: 2, loss: 1.15070474148
2016-10-26T02:56:53.259190556Z Epoch: 3, loss: 0.256429493427
2016-10-26T02:56:53.259195798Z Epoch: 4, loss: 0.0424121692777
2016-10-26T02:56:53.259201130Z Epoch: 5, loss: 0.00265768845566
2016-10-26T02:56:53.259206653Z Epoch: 6, loss: 0.000737804220989
2016-10-26T02:56:53.259211961Z Epoch: 7, loss: 0.00451849261299
2016-10-26T02:56:53.259217125Z Epoch: 8, loss: 0.0076722134836
2016-10-26T02:56:53.259222407Z Epoch: 9, loss: 0.00959475897253
2016-10-26T02:56:53.259227708Z [0:00:02.928687] End of standalone training.
2016-10-26T02:56:53.259235015Z Get the model, w: 1.88596236706, b: 9.95958137512
2016-10-26T02:56:53.259240345Z Exporting trained model to /tmp/linear_model/
2016-10-26T02:56:53.259246348Z Done exporting!
```

Practice: Bring Your Own Image

- Dependencies in other languages
- Build your own docker image
- cloudml jobs submit -d $DockerImage

Practice: Bring Your Own Image

- Dependencies in other languages
- Build your own docker image
- cloudml jobs submit -d $DockerImage

Other frameworks?

TensorFlow

scikit learn

K

theano

dmlc mxnet

dmlc XGBoost

Caffe

Microsoft CNTK

PaddlePaddle
Baidu Deep Learning

Other frameworks? Possible!

TensorFlow

scikit learn

K

theano

dmlc mxnet

dmlc XGBoost

Caffe

Microsoft CNTK

PaddlePaddle
Baidu Deep Learning

Practice: Model Zoo

- Image APIs
  - Inception
  - StyleTransfer
  - Img2Txt
  - Classification
- Voice APIs
  - Recognition
  - Analyses
  - WaveNet
- Text APIs
  - Translation
  - Segmentation
  - Chatbot
- Other APIs
  - Games AI
  - Search

Conclusion

- 机器学习与深度学习应用
- 深度学习平台架构与设计
- 深度学习平台应用与实践

Conclusion

- 机器学习与深度学习应用
- 深度学习平台架构与设计
- 深度学习平台应用与实践

Conclusion

Phase 1	Phase 2	Phase 3	Phase 4
云计算	大数据	深度学习	云深度学习

GIAC BEIJING Dec.12.16-17

技术架构未来

架構 ARCHNOTES 高可用架构

Thank you!

@tobe迪豪

(Next Page)

Thanks