深度学习

深度学习云服务介绍

陈迪豪

Agenda

- Cloud Machine Learning平台
- Cloud-ml服务介绍(PaaS)
- Cloud-ai服务介绍(SaaS)

Agenda

- Cloud Machine Learning平台
- Cloud-ml服务介绍(PaaS)
- Cloud-ai服务介绍(SaaS)

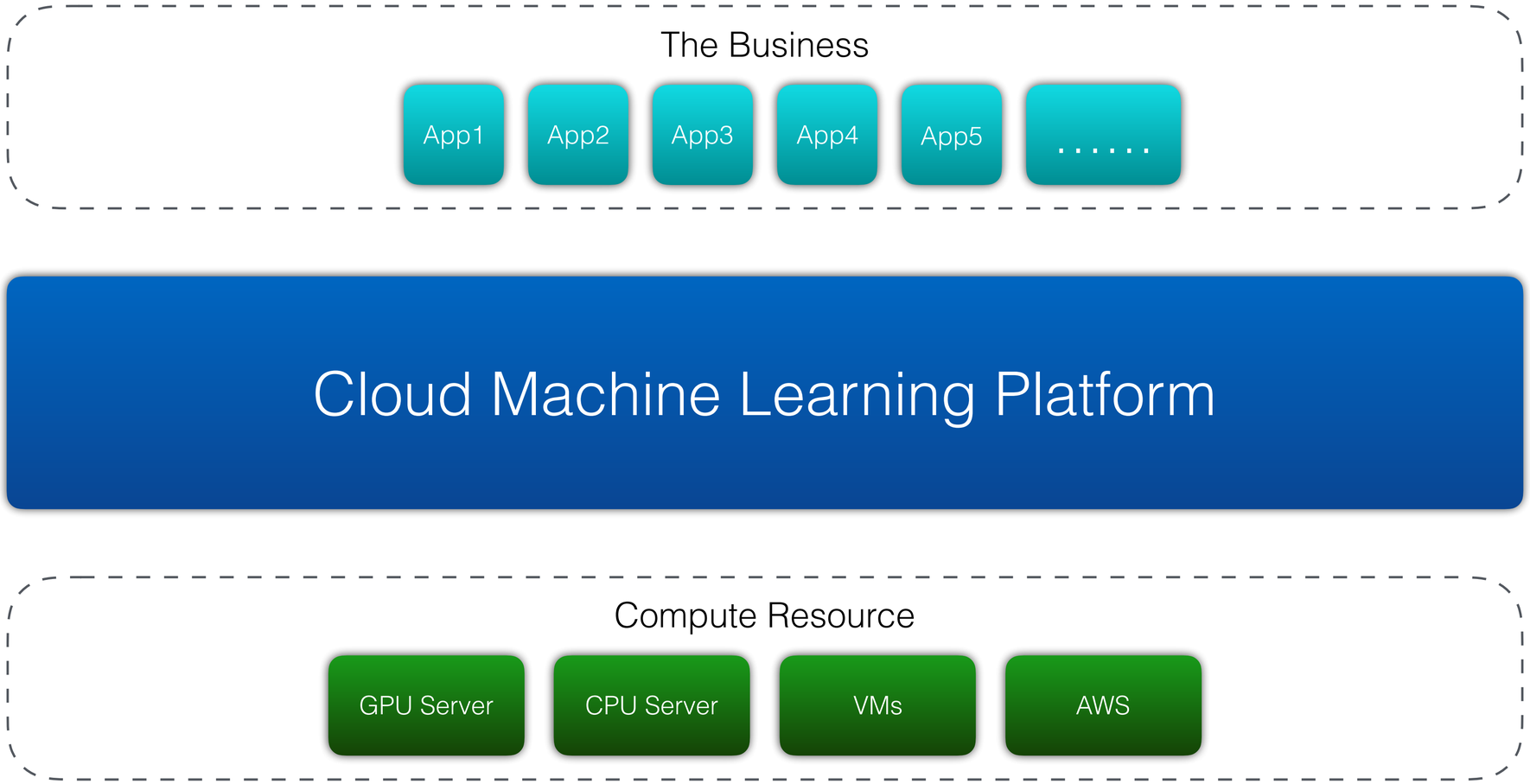
Cloud Machine Learning

Cloud Machine Learning

Cloud Machine Learning

Computation + Algorithm + "Big data"

Cloud Machine Learning



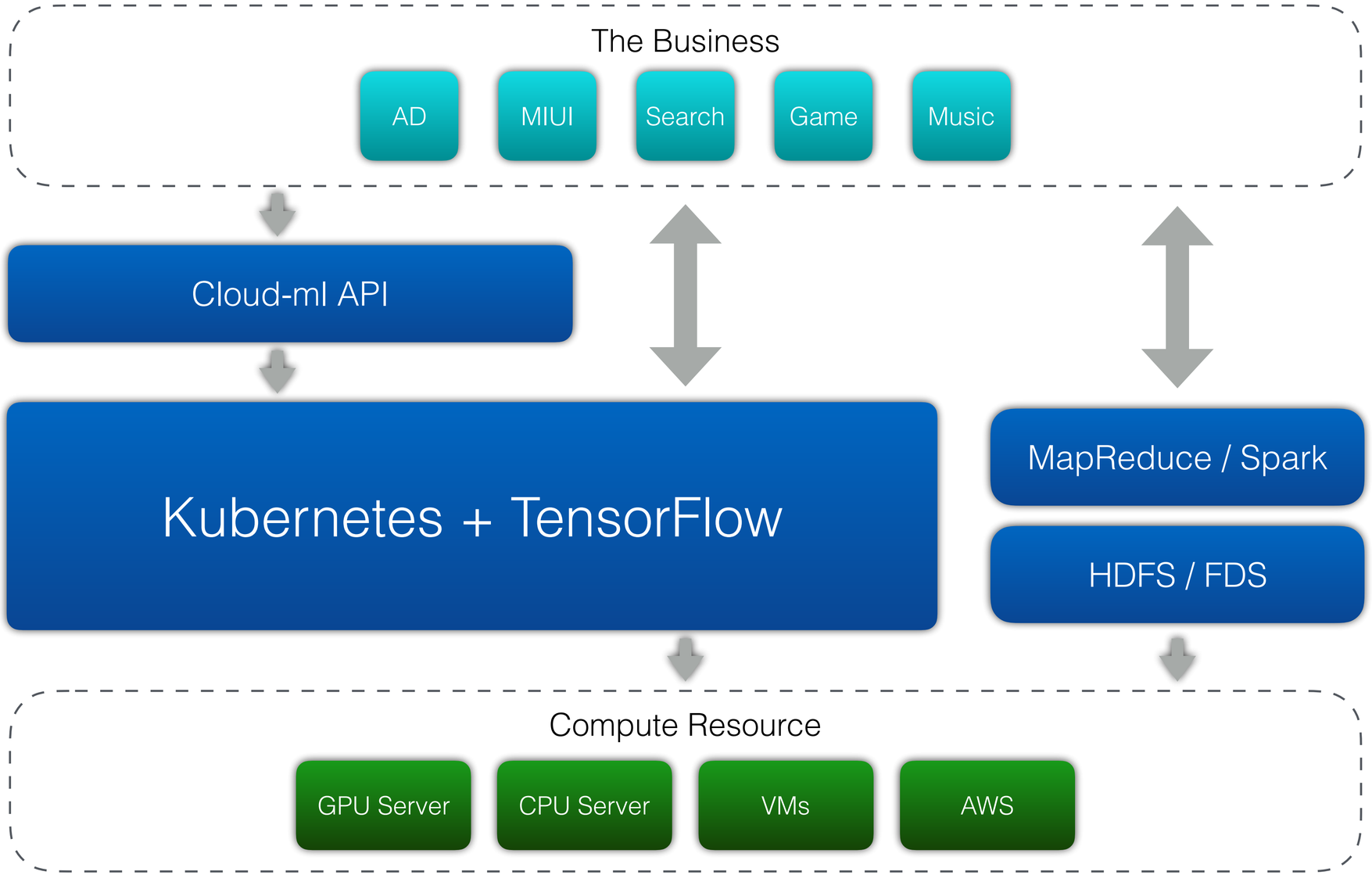
Cloud Machine Learning

- “无限”的存储和计算资源
- 缩短环境部署和启动时间
- 高利用率和低性能损耗
- 错误容忍和自动故障迁移
- 无人值守和自动化调参

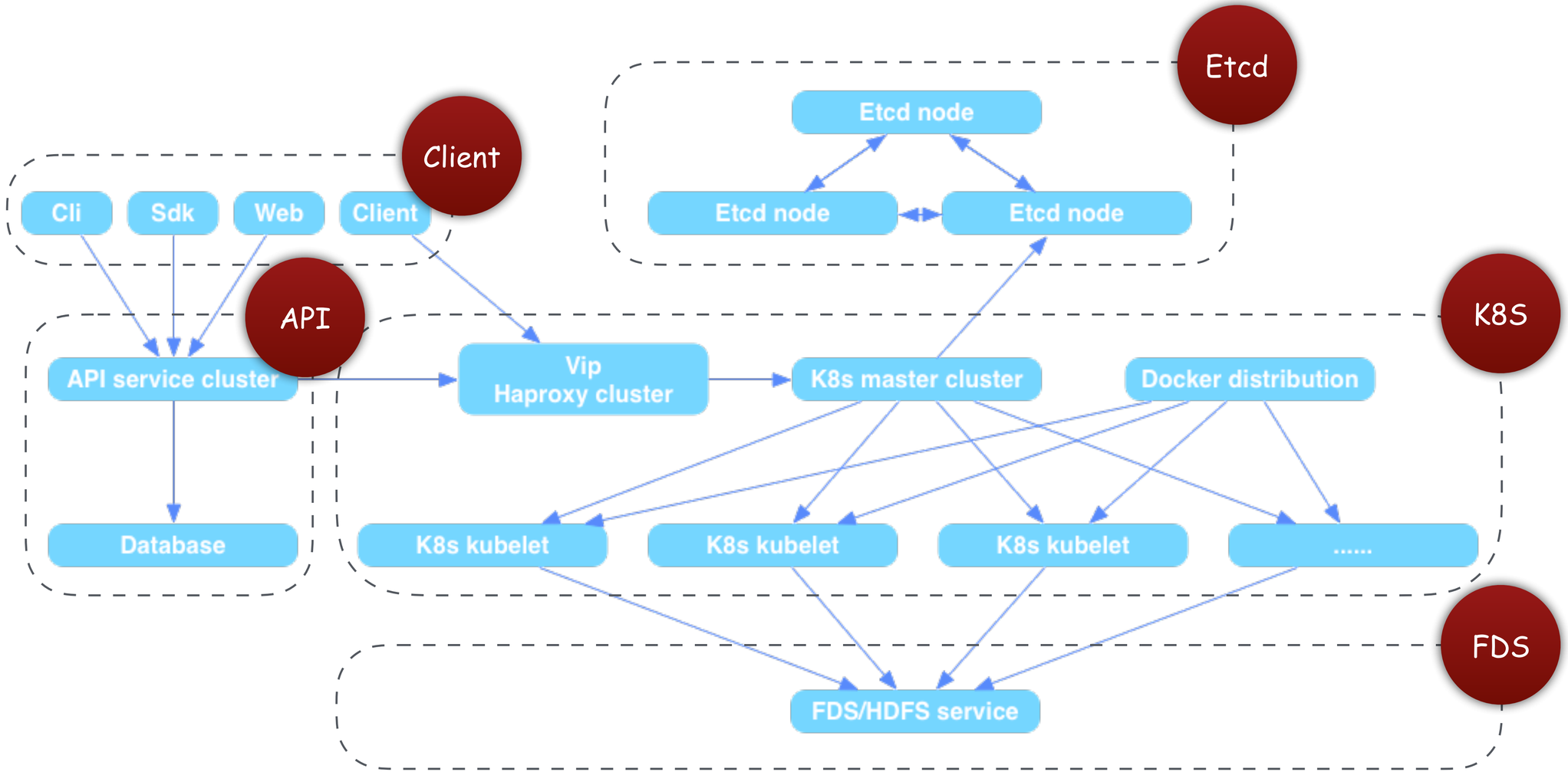
Agenda

- Cloud Machine Learning平台
- Cloud-ml服务介绍(PaaS)
- Cloud-ai服务介绍(SaaS)

Cloud-ml Service



Cloud-ml Service



Cloud-ml Service

Cloud-ml: TrainJob

API

SDK

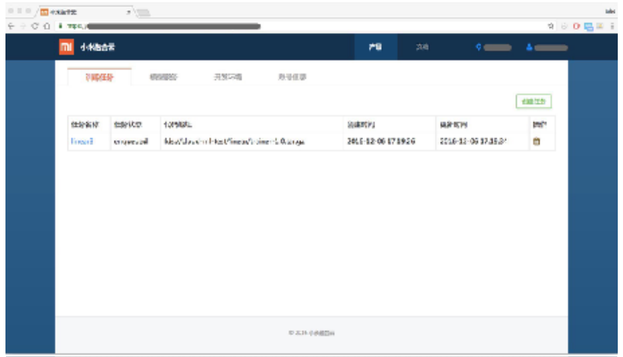
Command

Web

```
POST /cloud_ml/v1/train

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

```
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)
```

```
→ 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
```



Cloud-ml: TrainJob

```
cloudml jobs submit
    -n mnist
    -m trainer.task
    -u fds://cloud-ml/trainer-1.0.tar.gz
```

```python
import tensorflow as tf
import numpy as np

# Prepare train data
train_X = np.linspace(-1, 1, 100)
train_Y = 2 * train_X + np.random.randn(*train_X.shape) * 0.33 + 10

# Define the model
X = tf.placeholder("float")
Y = tf.placeholder("float")
w = tf.Variable(0.0, name="weight")
b = tf.Variable(0.0, name="bias")
loss = tf.square(Y - tf.mul(X, w) - b)
train_op = tf.train.GradientDescentOptimizer(0.01).minimize(loss)

# Create session to run
with tf.Session() as sess:
    sess.run(tf.initialize_all_variables())

    epoch = 1
    for i in range(100):
        for (x, y) in zip(train_X, train_Y):
            _, w_value, b_value = sess.run([train_op, w, b],
                                           feed_dict={X: x,
                                                      Y: y})
        print("Epoch: {}, w: {}, b: {}".format(epoch, w_value, b_value))
        epoch += 1
```

Cloud-ml: TrainJob

```
cloudml jobs submit
    -n mnist
    -m trainer.task
    -u fds://cloud-ml/trainer-1.0.tar.gz
    --cpu_limit 56
    --memory_limit 128G
    --gpu_limit 4
```

```python
import tensorflow as tf
import numpy as np

# Prepare train data
train_X = np.linspace(-1, 1, 100)
train_Y = 2 * train_X + np.random.randn(*train_X.shape) * 0.33 + 10

# Define the model
X = tf.placeholder("float")
Y = tf.placeholder("float")
w = tf.Variable(0.0, name="weight")
b = tf.Variable(0.0, name="bias")
loss = tf.square(Y - tf.mul(X, w) - b)
train_op = tf.train.GradientDescentOptimizer(0.01).minimize(loss)

# Create session to run
with tf.Session() as sess:
    sess.run(tf.initialize_all_variables())

    epoch = 1
    for i in range(100):
        for (x, y) in zip(train_X, train_Y):
            _, w_value, b_value = sess.run([train_op, w, b],
                                           feed_dict={X: x,
                                                      Y: y})
        print("Epoch: {}, w: {}, b: {}".format(epoch, w_value, b_value))
        epoch += 1
```

Cloud-ml: TrainJob

```
cloudml jobs submit
    -n mnist
    -m trainer.task
    -u fds://cloud-ml/trainer-1.0.tar.gz
    --ps_count 8
    --worker_count 16
```

```python
import tensorflow as tf
import numpy as np

# Prepare train data
train_X = np.linspace(-1, 1, 100)
train_Y = 2 * train_X + np.random.randn(*train_X.shape) * 0.33 + 10

# Define the model
X = tf.placeholder("float")
Y = tf.placeholder("float")
w = tf.Variable(0.0, name="weight")
b = tf.Variable(0.0, name="bias")
loss = tf.square(Y - tf.mul(X, w) - b)
train_op = tf.train.GradientDescentOptimizer(0.01).minimize(loss)

# Create session to run
with tf.Session() as sess:
    sess.run(tf.initialize_all_variables())

    epoch = 1
    for i in range(100):
        for (x, y) in zip(train_X, train_Y):
            _, w_value, b_value = sess.run([train_op, w, b],
                                           feed_dict={X: x,
                                                      Y: y})
        print("Epoch: {}, w: {}, b: {}".format(epoch, w_value, b_value))
        epoch += 1
```

Cloud-ml: TrainJob

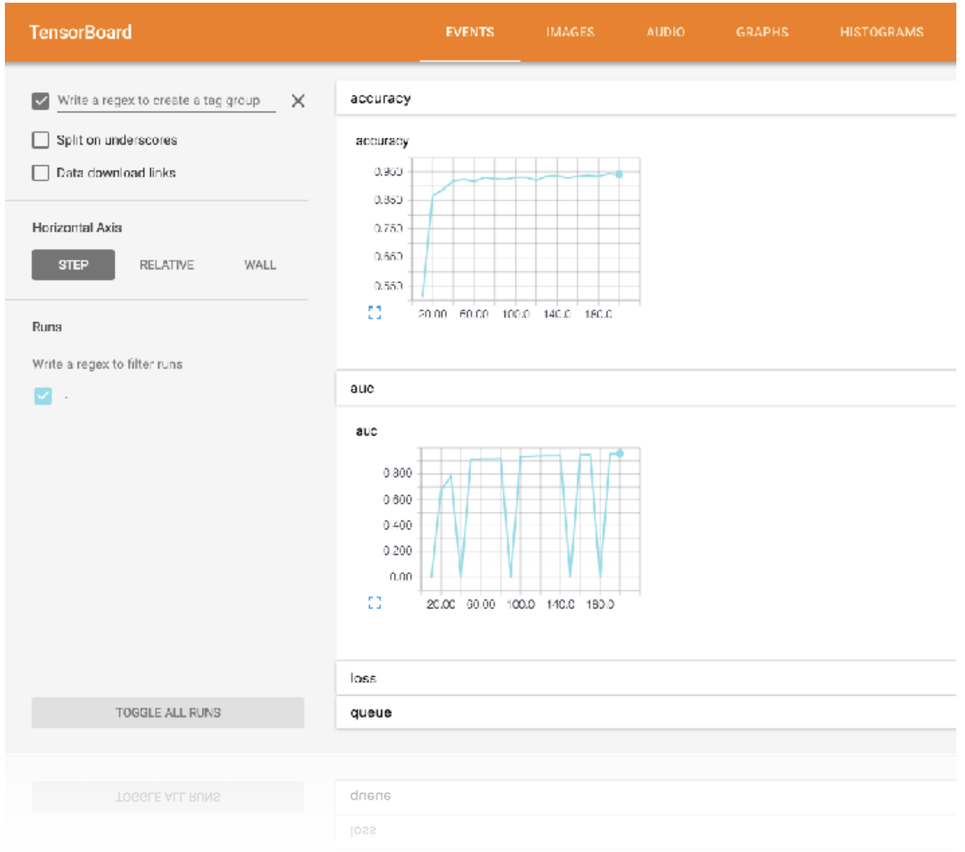
```
cloudml jobs submit
    -n mnist
    -m trainer.task
    -u fds://cloud-ml/trainer-1.0.tar.gz
    --framework mxnet
    --framework_version 0.9.0
```

```python
import mxnet as mx
import argparse
import os, sys
import train_model

def get_loc(data, attr={'lr_mult':'0.01'}):
    """
    the localisation network in lenet-stn, it will increase acc about more than 1%,
    when num-epoch >=15
    """
    loc = mx.symbol.Convolution(data=data, num_filter=30, kernel=(5, 5), stride=(2,2))
    loc = mx.symbol.Activation(data = loc, act_type='relu')
    loc = mx.symbol.Pooling(data=loc, kernel=(2, 2), stride=(2, 2), pool_type='max')
    loc = mx.symbol.Convolution(data=loc, num_filter=60, kernel=(3, 3), stride=(1,1), pad=(1, 1))
    loc = mx.symbol.Activation(data = loc, act_type='relu')
    loc = mx.symbol.Pooling(data=loc, global_pool=True, kernel=(2, 2), pool_type='avg')
    loc = mx.symbol.Flatten(data=loc)
    loc = mx.symbol.FullyConnected(data=loc, num_hidden=6, name="stn_loc", attr=attr)
    return loc

def get_mlp():
    """
    multi-layer perceptron
    """
    data = mx.symbol.Variable('data')
    fc1  = mx.symbol.FullyConnected(data = data, name='fc1', num_hidden=128)
    act1 = mx.symbol.Activation(data = fc1, name='relu1', act_type="relu")
    fc2  = mx.symbol.FullyConnected(data = act1, name = 'fc2', num_hidden = 64)
    act2 = mx.symbol.Activation(data = fc2, name='relu2', act_type="relu")
    fc3  = mx.symbol.FullyConnected(data = act2, name='fc3', num_hidden=10)
    mlp  = mx.symbol.SoftmaxOutput(data = fc3, name = 'softmax')
    return mlp
```

Cloud-ml: TrainJob

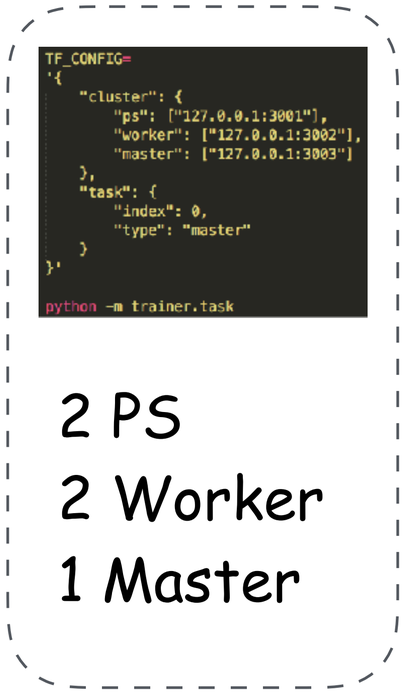
```
2016-10-26T02:56:47.561916526Z Processing trainer-1.0-py2.7.egg
2016-10-26T02:56:47.562317017Z Copying trainer-1.0-py2.7.egg to /usr/local/lib/python
2016-10-26T02:56:47.563445769Z Adding trainer 1.0 to easy-install.pth file
2016-10-26T02:56:47.564566658Z
2016-10-26T02:56:47.564574540Z Installed /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!
```

训练日志



模型指标

Cloud-ml: TrainJob




```
{
  "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"]
}
```

Cloud-ml: TrainJob

- ✤ 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
```

Cloud-ml: TrainJob

- ✤ Compatible with Google CloudML
- ✤ https://github.com/GoogleCloudPlatform/cloudml-samples

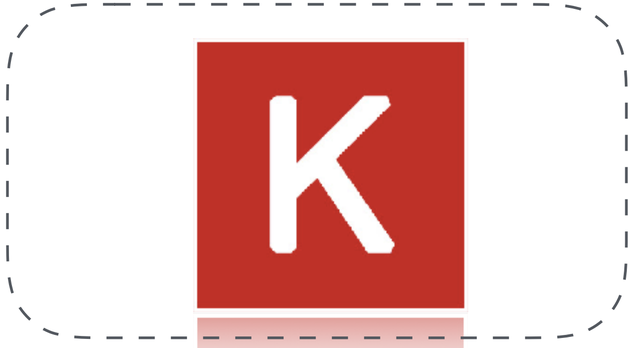
```
➜ trainable git:(master) ✗ python setup.py sdist --format=gztar
running sdist
running egg_info
creating trainer.egg-info
writing trainer.egg-info/PKG-INFO
writing top-level names to trainer.egg-info/top_level.txt
writing dependency_links to trainer.egg-info/dependency_links.txt
writing manifest file 'trainer.egg-info/SOURCES.txt'
reading manifest file 'trainer.egg-info/SOURCES.txt'
writing manifest file 'trainer.egg-info/SOURCES.txt'
warning: sdist: standard file not found: should have one of README, README.rst, README.txt

running check
warning: check: missing required meta-data: url

warning: check: missing meta-data: either (author and author_email) or (maintainer and maintainer_email) must be supplied

creating trainer-0.0.0
creating trainer-0.0.0/trainer
creating trainer-0.0.0/trainer.egg-info
copying files to trainer-0.0.0...
copying setup.py -> trainer-0.0.0
copying trainer/__init__.py -> trainer-0.0.0/trainer
copying trainer/task.py -> trainer-0.0.0/trainer
copying trainer.egg-info/PKG-INFO -> trainer-0.0.0/trainer.egg-info
copying trainer.egg-info/SOURCES.txt -> trainer-0.0.0/trainer.egg-info
copying trainer.egg-info/dependency_links.txt -> trainer-0.0.0/trainer.egg-info
copying trainer.egg-info/top_level.txt -> trainer-0.0.0/trainer.egg-info
Writing trainer-0.0.0/setup.cfg
creating dist
Creating tar archive
removing 'trainer-0.0.0' (and everything under it)
➜ trainable git:(master) ✗ ls dist
trainer-0.0.0.tar.gz
```

```
gcloud ml-engine jobs submit training
    mnist1
    --module-name trainer.task
    --package-path trainer/
```

or

```
cloudml jobs submit
    -n mnist1
    -m trainer.task
    -u fds://cloud-ml/trainer-1.0.0.tar.gz
```

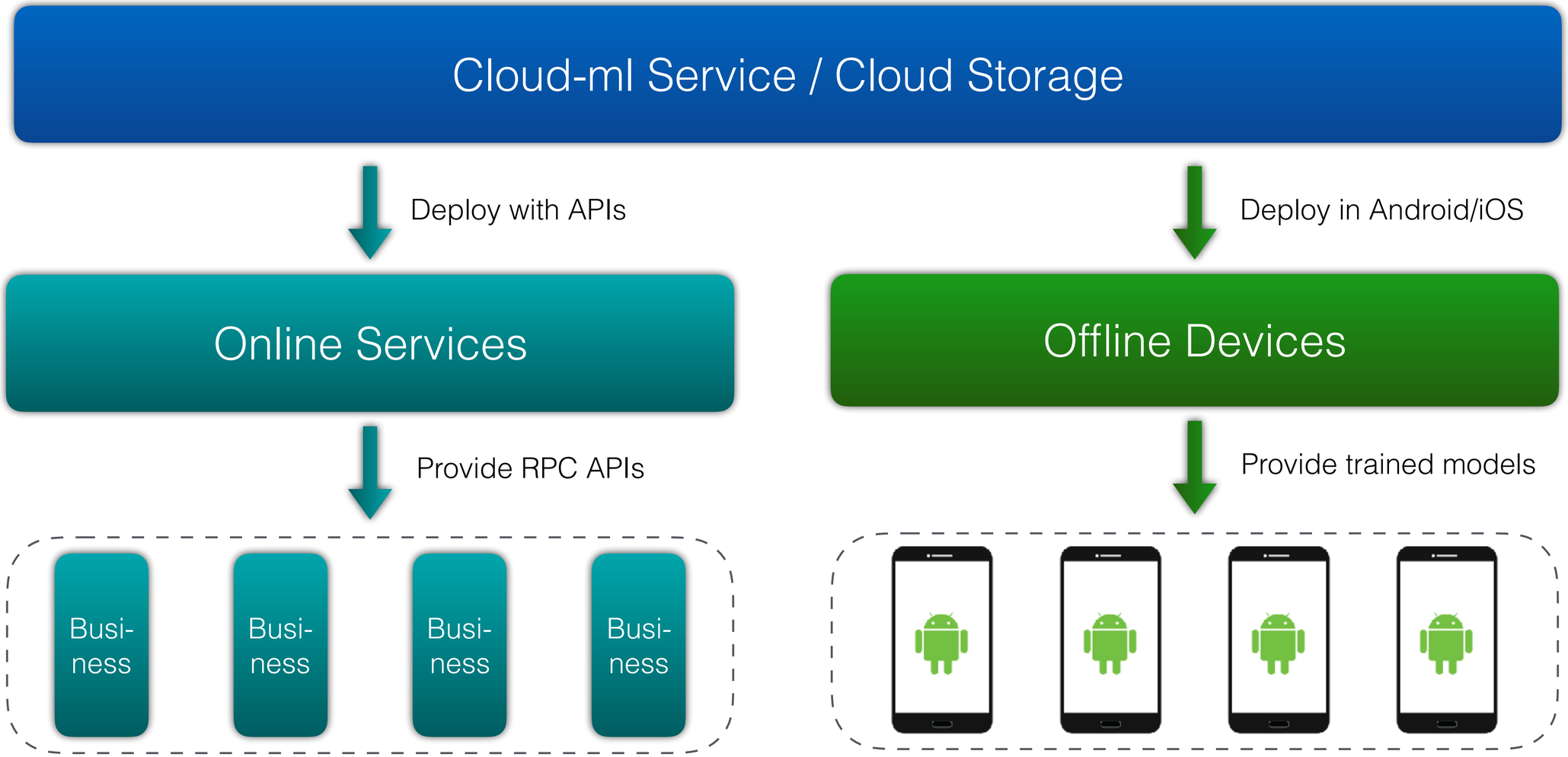
Cloud-ml: TrainJob



Cloud-ml: TrainJob

Framework	Version	Framework	Version
TensorFlow	0.12.0	Scikit-learn	0.17.0
	1.0.0	XGBoost	0.6.0
MXNet	0.9.0	PaddlePaddle	0.9.0
Theano	0.8.2	Gym	0.7.3
CNTK	2.0.0	Neon	1.8.2
Torch	master	Chainer	1.21.0
Caffe	0.9.0	PyTorch	0.1.10
Keras	1.1.1	Deeplearning4j	0.5.0
Lasagne	0.2.0	Dsstne	master

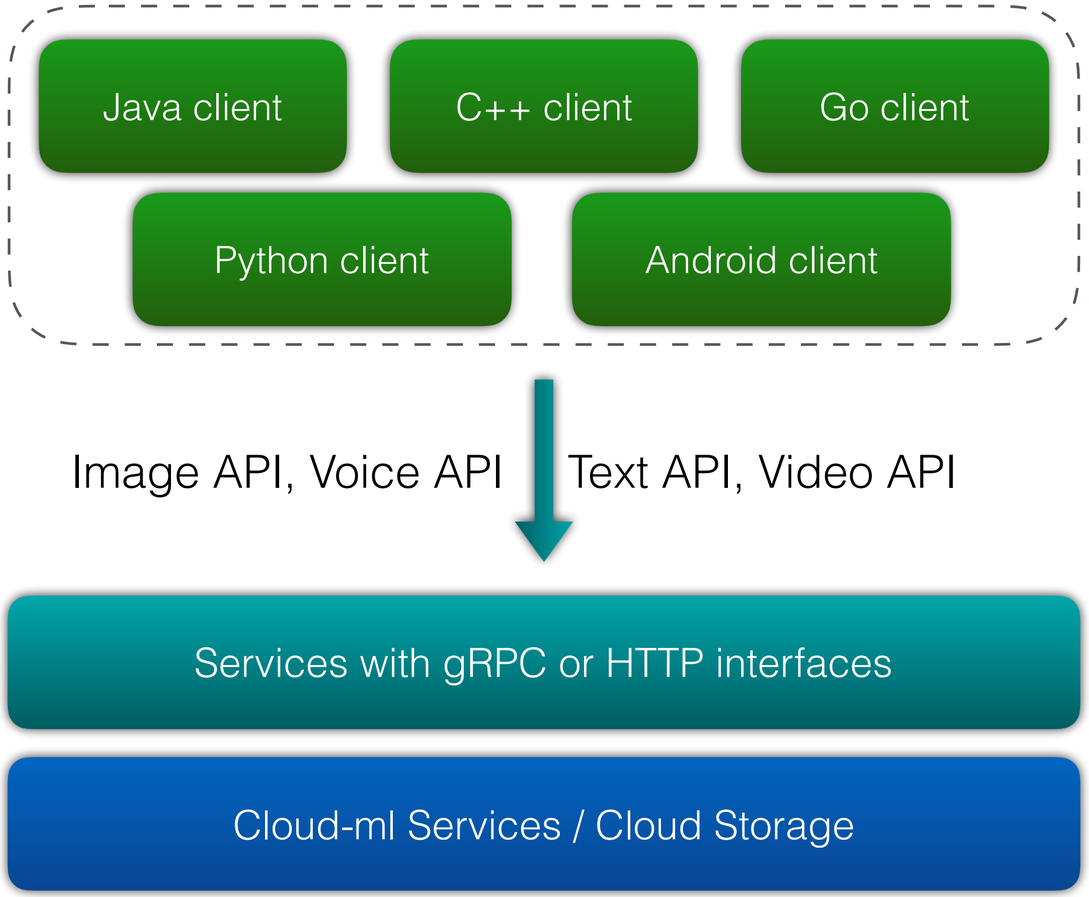
Cloud-ml: ModelService



Cloud-ml: ModelService

- Standard gRPC interfaces
- Support online model upgrade
- Support serving multiple versions
- Support replicas and load balancer

Cloud-ml: ModelService




```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: ModelService

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

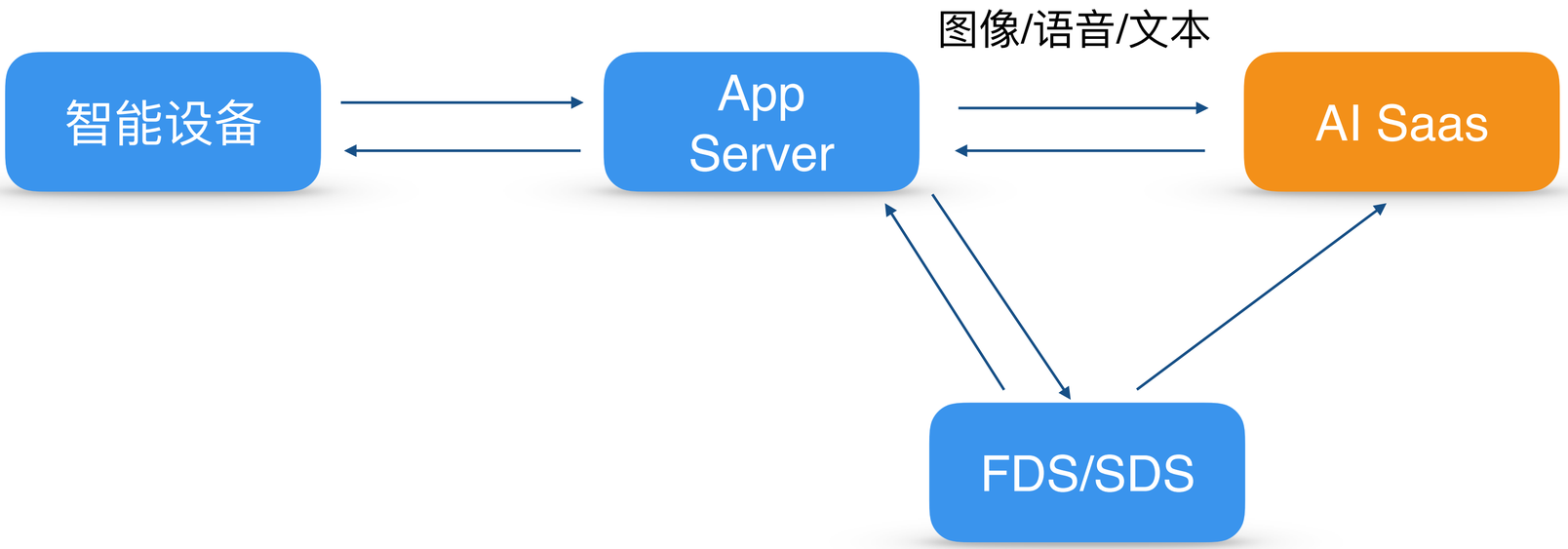
Agenda

- Cloud Machine Learning平台
- Cloud-ml服务介绍(PaaS)
- Cloud-ai服务介绍(SaaS)

SaaS服务

SaaS服务

- **应用场景**

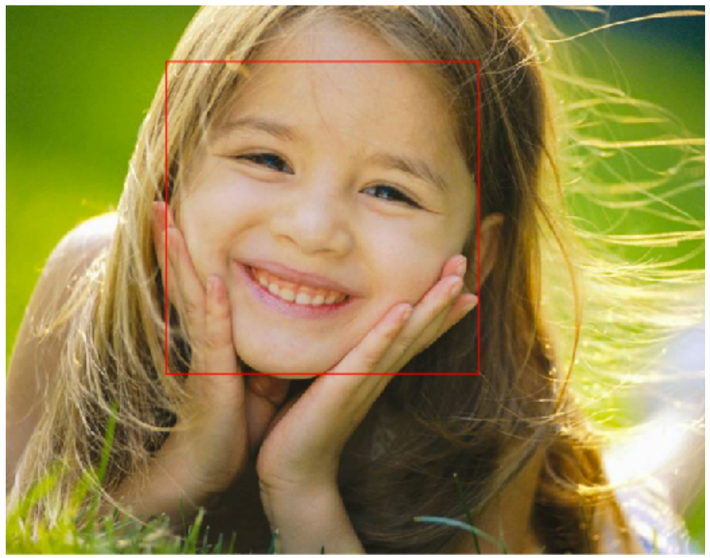


SaaS服务

图像识别

人脸检测: 人脸位置、性别、年龄

物体识别: 1500+ 物体分类(包括客厅、卧室等场景)



FaceInfo:
- topX: 208
- topY: 73
- width: 403
- height: 403
- child
- female
- age: 5.6

SaaS服务

图像识别

人脸检测: 人脸位置、性别、年龄

物体识别: 1500+ 物体分类(包括客厅、卧室等场景)

物体	置信度
客厅(living room)	0.52
餐厅(dining room)	0.14
大厅(hall)	0.08
休闲室(waiting room)	0.06

SaaS服务

- 图像识别

```
Credential credential = new Credential("YOUR-AK", "YOUR-SK");
VisionConfig config = new VisionConfig("cnbj3.vision.api.xiaomi.com");
GalaxyVisionClient visionClient = new GalaxyVisionClient(credential, config);
Image image = new Image();
byte[] data = IOUtils.loadImage("path-to-face-image");
image.setContent(data);
// Alternatively, you can specify the image by fds uri as:
// image.setUri("fds://cnbj1-fds.api.xiaomi.net/tst-team-2/img_930.jpg");

// send detect faces request
DetectFacesRequest facesRequest = new DetectFacesRequest();
facesRequest.setImage(image);
DetectFacesResult result = visionClient.detectFaces(facesRequest);
System.out.println("faces result: " + new Gson().toJson(result));

// send detect labels request
DetectLabelsRequest labelsRequest = new DetectLabelsRequest();
labelsRequest.setImage(image);
DetectLabelsResult labelsResult = visionClient.detectLabels(labelsRequest);
System.out.println("labels result: " + new Gson().toJson(labelsResult));
```

SaaS服务

图像识别	7月: 人脸对比、人脸识别、OCR、物体识别多分类
语音识别	语音识别、声纹识别、声纹对比
自然语言处理	聊天机器人

Conclusion

- Cloud Machine Learning平台
- Cloud-ml服务介绍(PaaS)
- Cloud-ai服务介绍(SaaS)

谢谢