七牛云 QINIU | IT大咖说 知识共享平台

“2007-2017” ECUG 十年高峰盛会

AI算法实现和云平台应用

2017.12.17

About Me

- HBase/Hadoop contributor
- OpenStack contributor
- Seagull(Docker Web UI) author
- TensorFlow/Kubernetes developer
- System architect at 4Paradigm

Agenda

- 人工智能与机器学习介绍
- 机器学习算法原理与实现
- 云机器学习平台架构实践

Agenda

- 人工智能与机器学习介绍
- 机器学习算法原理与实现
- 云机器学习平台架构实践

人工智能？

- 机械自动化 -> 人工智能自动工场
- 字符串生成 -> 人工智能自动编程
- 验证码识别 -> 人工智能黑客技术

```
array([[[     0,      1,      2, ...,    997,    998,    999],
        [  1000,   1001,   1002, ...,   1997,   1998,   1999],
        [  2000,   2001,   2002, ...,   2997,   2998,   2999],
        ...,
        [ 97000,  97001,  97002, ...,  97997,  97998,  97999],
        [ 98000,  98001,  98002, ...,  98997,  98998,  98999],
        [ 99000,  99001,  99002, ...,  99997,  99998,  99999]],

       [[100000, 100001, 100002, ..., 100997, 100998, 100999],
        [101000, 101001, 101002, ..., 101997, 101998, 101999],
        [102000, 102001, 102002, ..., 102997, 102998, 102999],
        ...,
        [197000, 197001, 197002, ..., 197997, 197998, 197999],
        [198000, 198001, 198002, ..., 198997, 198998, 198999],
        [199000, 199001, 199002, ..., 199997, 199998, 199999]],

       [[200000, 200001, 200002, ..., 200997, 200998, 200999],
        [201000, 201001, 201002, ..., 201997, 201998, 201999],
        [202000, 202001, 202002, ..., 202997, 202998, 202999],
        ...,
        [297000, 297001, 297002, ..., 297997, 297998, 297999],
        [298000, 298001, 298002, ..., 298997, 298998, 298999],
        [299000, 299001, 299002, ..., 299997, 299998, 299999]],

       ...,
       [[9700000, 9700001, 9700002, ..., 9700997, 9700998, 9700999],
        [9701000, 9701001, 9701002, ..., 9701997, 9701998, 9701999],
        [9702000, 9702001, 9702002, ..., 9702997, 9702998, 9702999],
        ...,
        [9797000, 9797001, 9797002, ..., 9797997, 9797998, 9797999],
        [9798000, 9798001, 9798002, ..., 9798997, 9798998, 9798999],
        [9799000, 9799001, 9799002, ..., 9799997, 9799998, 9799999]],

       [[9800000, 9800001, 9800002, ..., 9800997, 9800998, 9800999],
        [9801000, 9801001, 9801002, ..., 9801997, 9801998, 9801999],
        [9802000, 9802001, 9802002, ..., 9802997, 9802998, 9802999],
        ...,
        [9897000, 9897001, 9897002, ..., 9897997, 9897998, 9897999],
        [9898000, 9898001, 9898002, ..., 9898997, 9898998, 9898999],
        [9899000, 9899001, 9899002, ..., 9899997, 9899998, 9899999]],

       [[9900000, 9900001, 9900002, ..., 9900997, 9900998, 9900999],
        [9901000, 9901001, 9901002, ..., 9901997, 9901998, 9901999],
        [9902000, 9902001, 9902002, ..., 9902997, 9902998, 9902999],
        ...,
        [9997000, 9997001, 9997002, ..., 9997997, 9997998, 9997999],
        [9998000, 9998001, 9998002, ..., 9998997, 9998998, 9998999],
        [9999000, 9999001, 9999002, ..., 9999997, 9999998, 9999999]]])
```

机器学习介绍

- "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E." - Mitchell

- 场景 + 指标 + 样本 -> 机器学习模型

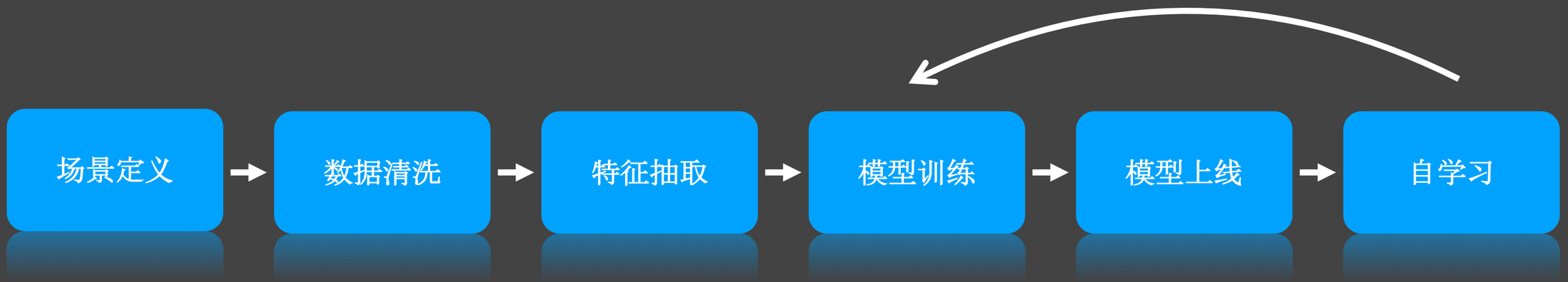
机器学习应用

- 1000张猫的图片，得到一个识别猫的模型
- 1000万盘围棋棋局，得到一个AlphaGo模型
- 历史的信用卡操作记录，得到一个反欺诈模型
- ……

机器学习应用

- 1000张猫的图片，得到一个识别猫的模型
- 1000万盘围棋棋局，得到一个AlphaGo模型
- 所有的信用卡操作记录，得到一个反欺诈模型
- ……

得到机器学习应用



- 二分类
- 多分类
- 回归
- 聚类

- MR
- Spark
- KG

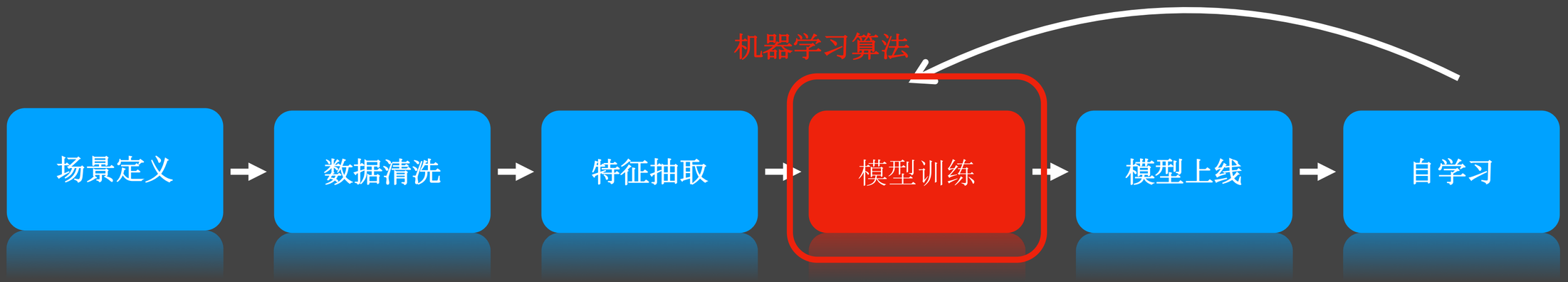
- DSL
- Parser
- Libsvm
- Tfrecord

- LR
- GBDT
- DNN
- HETree
- NB

- Thrift server
- Load balance
- Hight available
- AKSK

- SDK
- Adapter
- Online learning

得到机器学习应用



- 二分类
- 多分类
- 回归
- 聚类

- MR
- Spark
- KG

- DSL
- Parser
- Libsvm
- Tfrecord

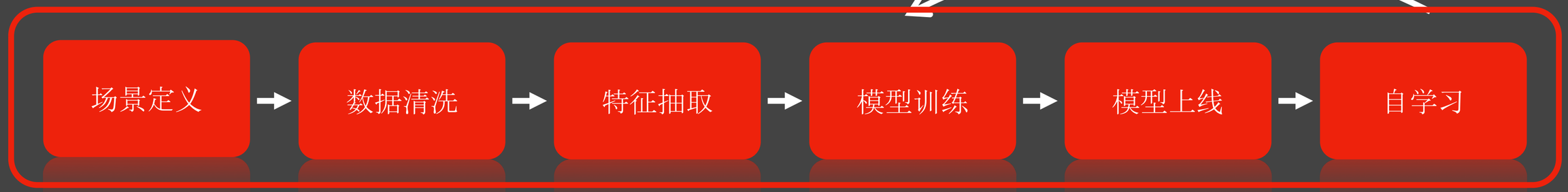
- LR
- GBDT
- DNN
- HETree
- NB

- Thrift server
- Load balance
- Hight available
- AKSK

- SDK
- Adapter
- Online learning

得到机器学习应用

机器学习平台



- 二分类
- 多分类
- 回归
- 聚类

- MR
- Spark
- KG

- DSL
- Parser
- Libsvm
- Tfrecord

- LR
- GBDT
- DNN
- HETree
- NB

- Thrift server
- Load balance
- Hight available
- AKSK

- SDK
- Adapter
- Online learning

Agenda

- 人工智能与机器学习介绍
- 机器学习算法原理与实现
- 云机器学习平台架构实践

机器学习算法 - 逻辑回归

逻辑回归: 简单而强大的机器学习算法，广泛用于推荐系统、CTR等场景

- 监督型模型
- 线性模型
- 高性能
- 可解析性强
- 可拓展性强

机器学习算法 - 逻辑回归

	性别:男	性别:女	年龄:0-30	年龄:30-60	年龄:60-90	年龄:80+	收入>50
样本1	1	0	0	1	0	0	1
样本2	0	1	0	0	0	1	0
...	...	...	...	...	...	...	...
LR模型	0.7	0.4	0.6	0.9	0.3	0.01	

机器学习算法 - 逻辑回归

	性别:男	性别:女	年龄:0-30	年龄:30-60	年龄:60-90	年龄:80+	收入>50
LR模型	0.7	0.4	0.6	0.9	0.3	0.01	
新样本	1	0	1	0	0	0	
计算得分	0.7	0	0.6	0	0	0	

activation(0.7 + 0.6) = 1

机器学习算法 - 逻辑回归

```python
import tensorflow as tf
import numpy as np
from tensorflow.examples.tutorials.mnist import input_data

# Train data
mnist = input_data.read_data_sets("./mnist/", one_hot=True)

# Hyper parameters
learning_rate = 0.1
batch_size = 10
train_epoch_number = 100
display_interval = 10

# The variables to compute
x = tf.placeholder(tf.float32, [None, 784])
y = tf.placeholder(tf.float32, [None, 10])
W = tf.Variable(tf.zeros([784, 10]), name="weight")
b = tf.Variable(tf.zeros([10]), name="bias")

pred = tf.nn.softmax(tf.matmul(x, W) + b)
loss = tf.reduce_mean(-tf.reduce_sum(y * tf.log(pred), reduction_indices=1))
train_op = tf.train.GradientDescentOptimizer(learning_rate).minimize(loss)

init_op = tf.initialize_all_variables()
```

init_op = tf.initialize_all_variables()

机器学习算法 - 逻辑回归

逻辑回归: 简单而强大的机器学习算法，广泛用于推荐系统、CTR等场景

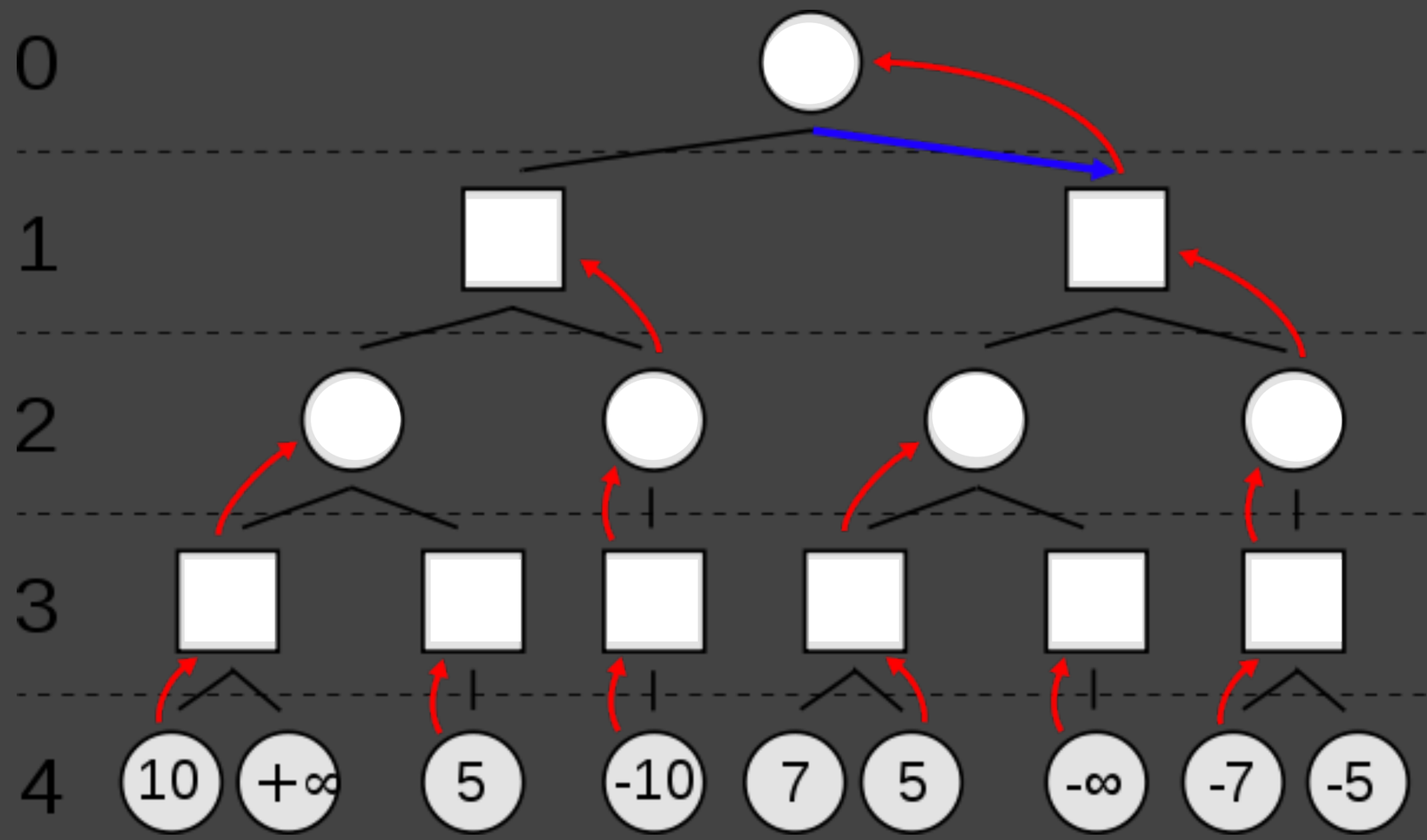
- 使用特征工程生成非线性Feature
- 模型为一维数组（长度与Feature维度相同）
- 定义Loss函数作为指标（如CrossEntropy）
- 使用梯度下降算法训练（如Adagrad优化器）
- 通过Parameter server实现分布式训练

机器学习算法 - AlphaZero

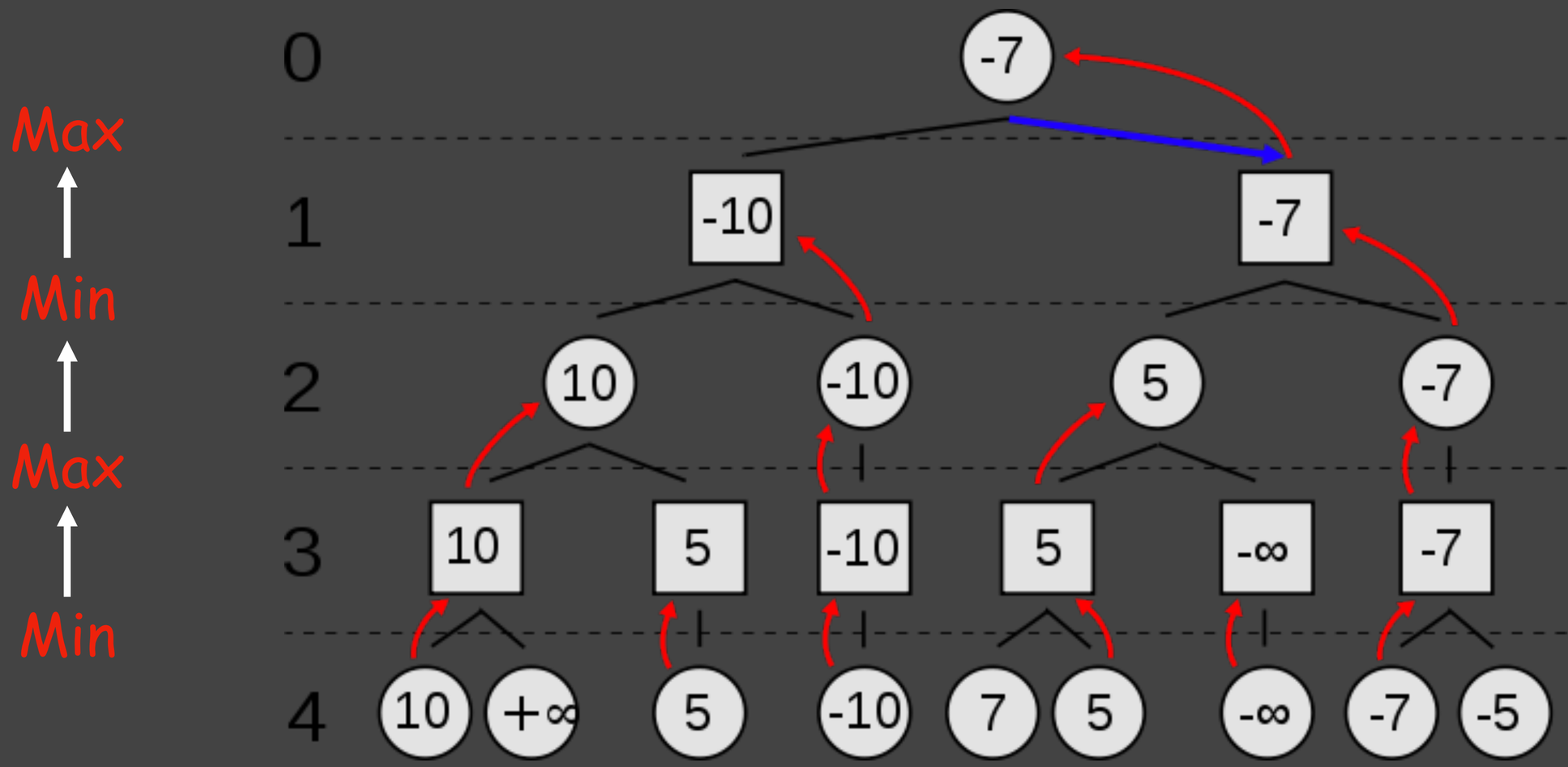
AlphaZero: 基于蒙特卡罗树搜索和神经网络的机器学习模型

- 基于蒙特卡罗树结构定义围棋规则（非自学习理解）
- 有监督学习，学习蒙特卡罗树搜索结果
- 使用Policy gradient（增强学习）生成更优的样本数据
- 模型为ResNet（输入Feature输出Probability）
- 适用于Combination game（零和、完全信息、无随机）

机器学习算法 - MinMax

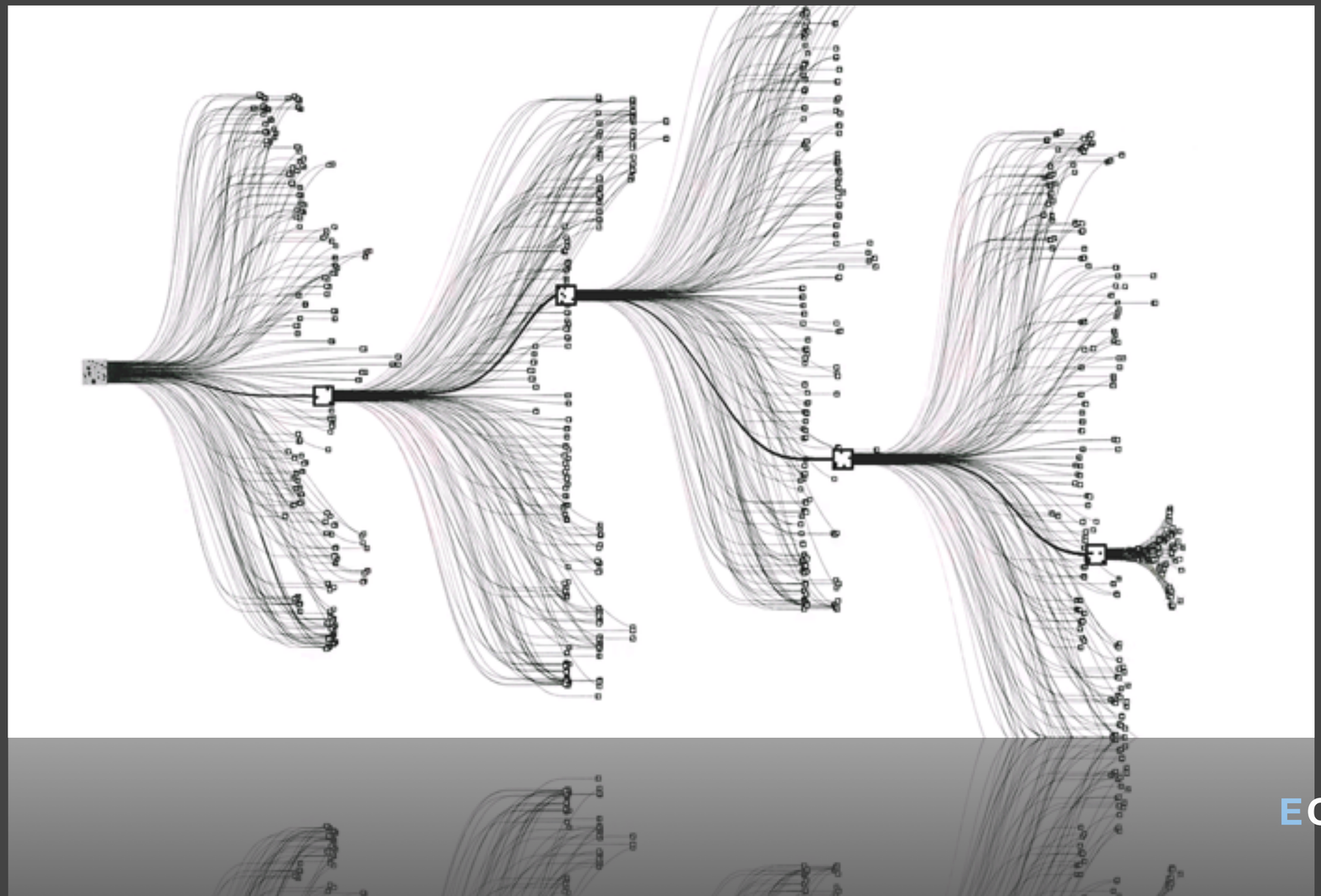


机器学习算法 - MinMax

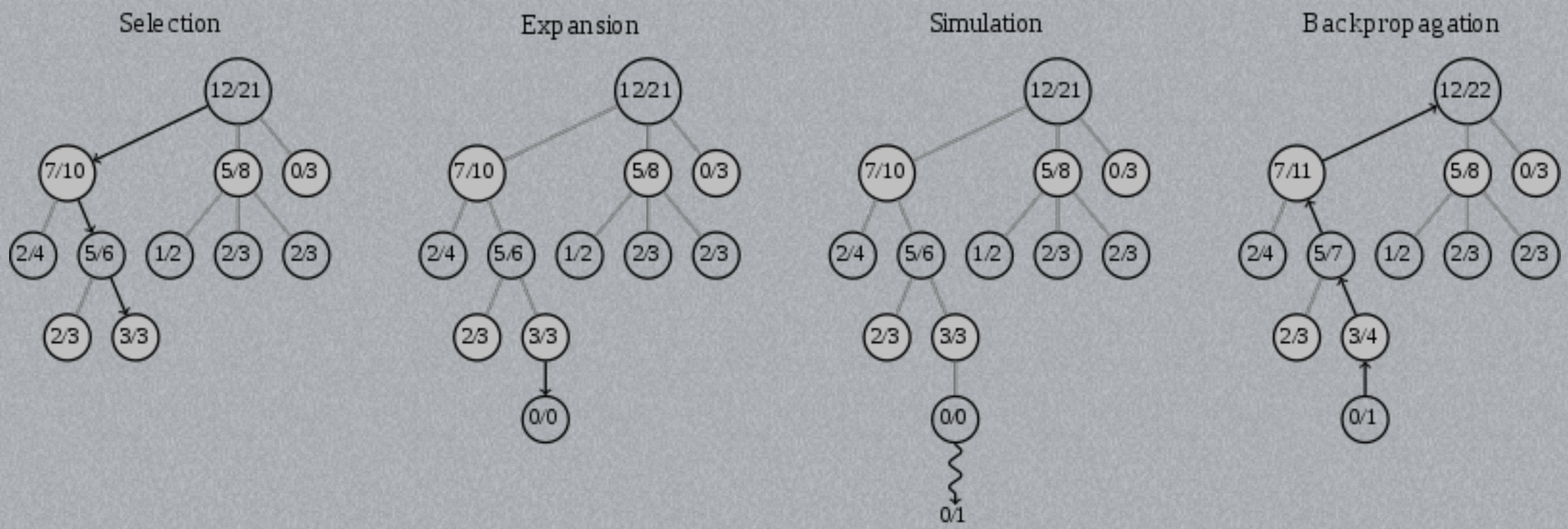


机器学习算法 - MCTS

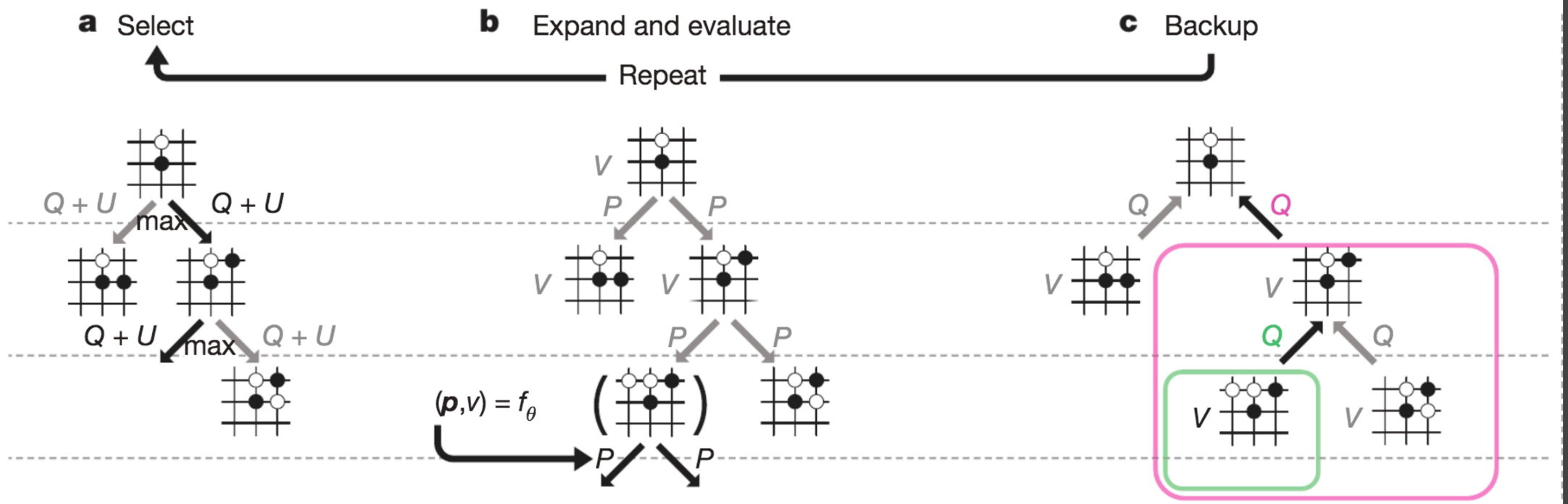
- Exploration
- Exploitation



机器学习算法 - MCTS


Selection
12/21
7/10
5/8
0/3
2/4
5/6
1/2
2/3
2/3
2/3
3/3
Expansion
12/21
7/10
5/8
0/3
2/4
5/6
1/2
2/3
2/3
2/3
3/3
0/0
Simulation
12/21
7/10
5/8
0/3
2/4
5/6
1/2
2/3
2/3
2/3
3/3
0/0
0/1
Backpropagation
12/22
7/11
5/8
0/3
2/4
5/7
1/2
2/3
2/3
2/3
3/4
0/1

机器学习算法 - AlphaZero



$$(\boldsymbol{p}, v) = f_\theta(s) \quad \text{and} \quad l = (z - v)^2 - \boldsymbol{\pi}^{\mathrm{T}} \log \boldsymbol{p} + c\|\theta\|^2$$

机器学习算法 - AlphaZero

- “UCB value”: Q + U
- Q: Sum_of_V(children) / N
- V: quality value, from neural network
- N: visit times
- U: C * P * “vanila UCB”
- C: Constant parameter to control exploration/exploitation
- P: probability to play, from neural network

机器学习算法 - AlphaZero

- “UCB value”: Q + U
- Q: Sum_of_V(children) / N
- V: quality value, from neural network
- N: visit times
- U: C * P * “vanila UCB”
- C: Constant parameter to control exploration/exploitation
- P: probability to play, from neural network

机器学习算法 - 自动求导

- Op求导（Add/Minus/Multiple/Divide/Square...）
- 链式法则
- 求偏导
- 梯度下降（Learning rate）
- Python operator override

机器学习算法 - 自动求导


```python
class AddOp(Op):
  """
  The addition operation which has only two inputs. The input can be
  primitive, ConstantOp, PlaceholerOp, VariableOp or other ops.
  """

  def __init__(self, input1, input2, name="Add"):
    super(AddOp, self).__init__(name)

    if not isinstance(input1, Op):
      self._op1 = ConstantOp(input1)
    else:
      self._op1 = input1

    if not isinstance(input2, Op):
      self._op2 = ConstantOp(input2)
    else:
      self._op2 = input2

    self._graph = graph.get_default_graph()
    self._graph.add_to_graph(self)

  def forward(self):
    result = self._op1.forward() + self._op2.forward()
    return result

  def grad(self, partial_derivative_opname=None):
    result = self._op1.grad(partial_derivative_opname) + self._op2.grad(
        partial_derivative_opname)
    return result
```

```python
class PowerOp(Op):
  def __init__(self, input, power, name="Power"):
    super(PowerOp, self).__init__(name)

    if not isinstance(input, Op):
      self._op = ConstantOp(input)
    else:
      self._op = input

    self._power = power

    self._graph = graph.get_default_graph()
    self._graph.add_to_graph(self)

  def forward(self):
    result = pow(self._op.forward(), self._power)
    return result

  def grad(self, partial_derivative_opname=None):
    if isinstance(self._op, PlaceholderOp) or isinstance(self._op, ConstantOp):
      # op is the constant
      grad = 0
    elif isinstance(self._op, VariableOp):
      # op is the variable
      grad = self._power * pow(self._op.forward(), self._power - 1)
    else:
      # op is other complex operation and use chain rule
      grad = self._power * pow(self._op.forward(), self._power - 1
                               ) * self._op.grad(partial_derivative_opname)

    return grad
```

机器学习算法 - 自动求导

basic_operation.py — tensorflow_examples

```python
#!/usr/bin/env python

# Copyright 2017 The Authors. All Rights Reserved.
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the Licens
# You may obtain a copy of the License at
#
#     http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, sof
# distributed under the License is distributed on an "AS IS" BAS
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express o
# See the License for the specific language governing permission
# limitations under the License.

import tensorflow as tf


def main():
  sess = tf.Session()

  # Add
  a = tf.constant(32.0)
  b = tf.constant(10.0)
  c = a + b
  print(sess.run(c))  # Should be 42.0

  # Minus
  c = a - b
  print(sess.run(c))  # Should be 22.0

  # Multiple
  c = a * b
  print(sess.run(c))  # Should be 320.0

  # Divide
  c = a / b
  print(sess.run(c))  # Should 3.2


if __name__ == "__main__":
  main()
```

basic_operation.py — miniflow_examples

```python
#!/usr/bin/env python

# Copyright 2017 The Authors. All Rights Reserved.
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the Licens
# You may obtain a copy of the License at
#
#     http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, sof
# distributed under the License is distributed on an "AS IS" BAS
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express o
# See the License for the specific language governing permission
# limitations under the License.

import miniflow as tf


def main():
  sess = tf.Session()

  # Add
  a = tf.constant(32.0)
  b = tf.constant(10.0)
  c = a + b
  print(sess.run(c))  # Should be 42.0

  # Minus
  c = a - b
  print(sess.run(c))  # Should be 22.0

  # Multiple
  c = a * b
  print(sess.run(c))  # Should be 320.0

  # Divide
  c = a / b
  print(sess.run(c))  # Should 3.2


if __name__ == "__main__":
  main()
```

机器学习算法 - 自动求导

- TensorFlow vs MiniFlow(22x speedup)
- 12.59 > 0.16 | 12.61 > 0.16 | 66.58 > 3.01

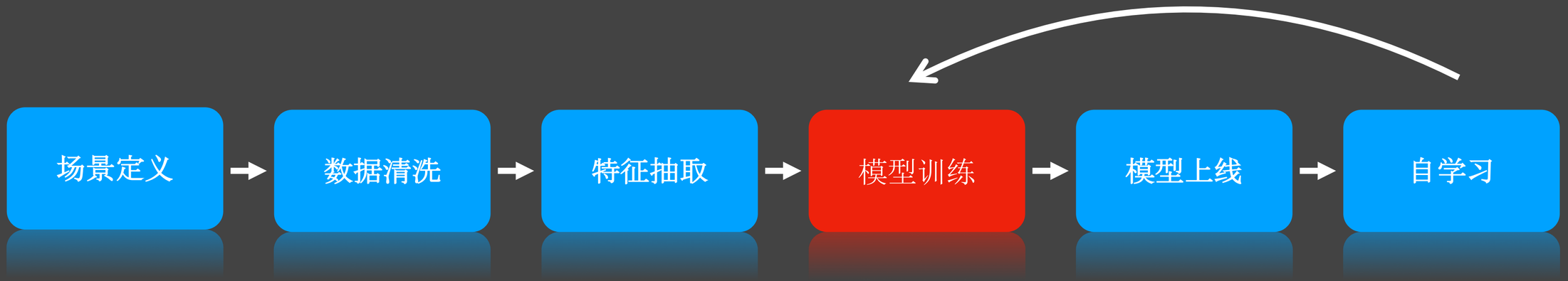
```
➜  add_operation git:(master) ✗ ./tensorflow_add.py
Benchmark scenario: add operation, epoch: 100000
Run time(s): 12.5948340893
➜  add_operation git:(master) ✗ cd ../multiple_operation
miniflow_multiple.py    tensorflow_multiple.py
➜  multiple_operation git:(master) ✗ ./tensorflow_multiple.py
Benchmark scenario: multiple operation, epoch: 100000
Run time(s): 12.6196570396
➜  multiple_operation git:(master) ✗ cd ../linear_regression
miniflow_linear_regression.py   tensorflow_linear_regression.py
➜  linear_regression git:(master) ✗ ./tensorflow_linear_regression.py
Benchmark scenario: linear regression, epoch: 100000
Run time(s): 66.5865750313
```

```
|➜  add_operation git:(master) ✗ ./miniflow_add.py
|Benchmark scenario: add operation, epoch: 100000
|Run time(s): 0.163687944412
|➜  add_operation git:(master) ✗ cd ../multiple_operation
|miniflow_multiple.py    tensorflow_multiple.py
|➜  multiple_operation git:(master) ✗ ./miniflow_multiple.py
|Benchmark scenario: multiple operation, epoch: 100000
|Run time(s): 0.165066957474
|➜  multiple_operation git:(master) ✗ cd ../linear_regression
|miniflow_linear_regression.py   tensorflow_linear_regression.py
|➜  linear_regression git:(master) ✗ ./miniflow_linear_regression.py
|Benchmark scenario: linear regression, epoch: 100000
|Run time(s): 3.01005196571
```

Agenda

- 人工智能与机器学习介绍
- 机器学习算法原理与实现
- 云机器学习平台架构实践

云机器学习平台 - 应用流程



- 二分类
- 多分类
- 回归
- 聚类

- MR
- Spark
- KG

- DSL
- Parser
- Libsvm
- Tfrecord

- LR
- GBDT
- DNN
- HETree
- NB

- Thrift server
- Load balance
- Hight available
- AKSK

- SDK
- Adapter
- Online learning

云机器学习平台 - 架构设计

- TensorFlow LR支持10亿维稀疏模型，10万亿维呢？
- 开源框架与自研框架（机器学习算法）的集成
- 异构计算集群（CPU、GPU、虚拟机、云平台）的支持
- 机器学习工作流的支持

云机器学习平台 - 架构设计

IaaS	PaaS	"MLaaS"
✤ AWS VMs	✤ Google CloudML	✤ Azure ML Studio
✤ AliCloud VMs	✤ Xiaomi CloudML	✤ 4Paradigm prophet
✤ AliCloud VMs	✤ AliCloud PAI	

图像处理 | 自然语言处理 | 语音处理 | 推荐系统

- 先知机器学习平台

API / SDK / WEB / Command-line

模型训练 | 模型服务 | 数据管理 | 特征抽取

Distributed TensorFlow | Distributed MXNet | 模型评估 | 自学习服务

Kubernetes Executor | Yarn / Hadoop Executor

CPU集群 | GPU集群 | FPGA集群 | 公有云 | 私有云 | 其他服务器

云机器学习平台 - "Google CloudML"

七牛云 QINIU | IT大咖说 知识共享平台

```
FROM ubuntu:16.04

RUN apt-get update && apt-get install -y --no-install-re
        build-essential \
        curl \
        libfreetype6-dev \
        libpng12-dev \
        libzmq3-dev \
        pkg-config \
        python \
        python-dev \
        rsync \
        software-properties-common \
        unzip \
        && \
    apt-get clean && \
    rm -rf /var/lib/apt/lists/*

RUN pip --no-cache-dir install \
    http://storage.googleapis.com/tensorflow/linux/cpu/t

COPY jupyter_notebook_config.py /root/.jupyter/
COPY notebooks /notebooks
COPY run_jupyter.sh /

EXPOSE 6006
EXPOSE 8888

WORKDIR "/notebooks"

CMD ["/run_jupyter.sh", "--allow-root"]
```

Step 1: Build docker image

```
@csrf_exempt
def train(request):
  if request.method == 'POST':

    # Get request data
    body = json.loads(request.body)
    job_name = body.get('job_name')
    job_module_name = body.get('module_name')
    job_trainer_uri = body.get('trainer_uri')
    job_args = body.get('job_args')
    job_train_data_paths = body.get('trainer_data_paths')
    job_eval_data_paths = body.get('job_eval_data_paths')
    job_output_path = body.get('output_path')
    job_master_spec = body.get('master_spec')

    # Generate Kuberenetes resources
    time_suffix = datetime.datetime.now().strftime("%m%d%H%M")
    kubernetes_resource_name = job_name + "-" + time_suffix
    train_container_run_commands = ["/run_user_module.py",
      job_trainer_uri, job_module_name]

    client = KubernetesClient().get_client()
```

Step 2: Implement API service

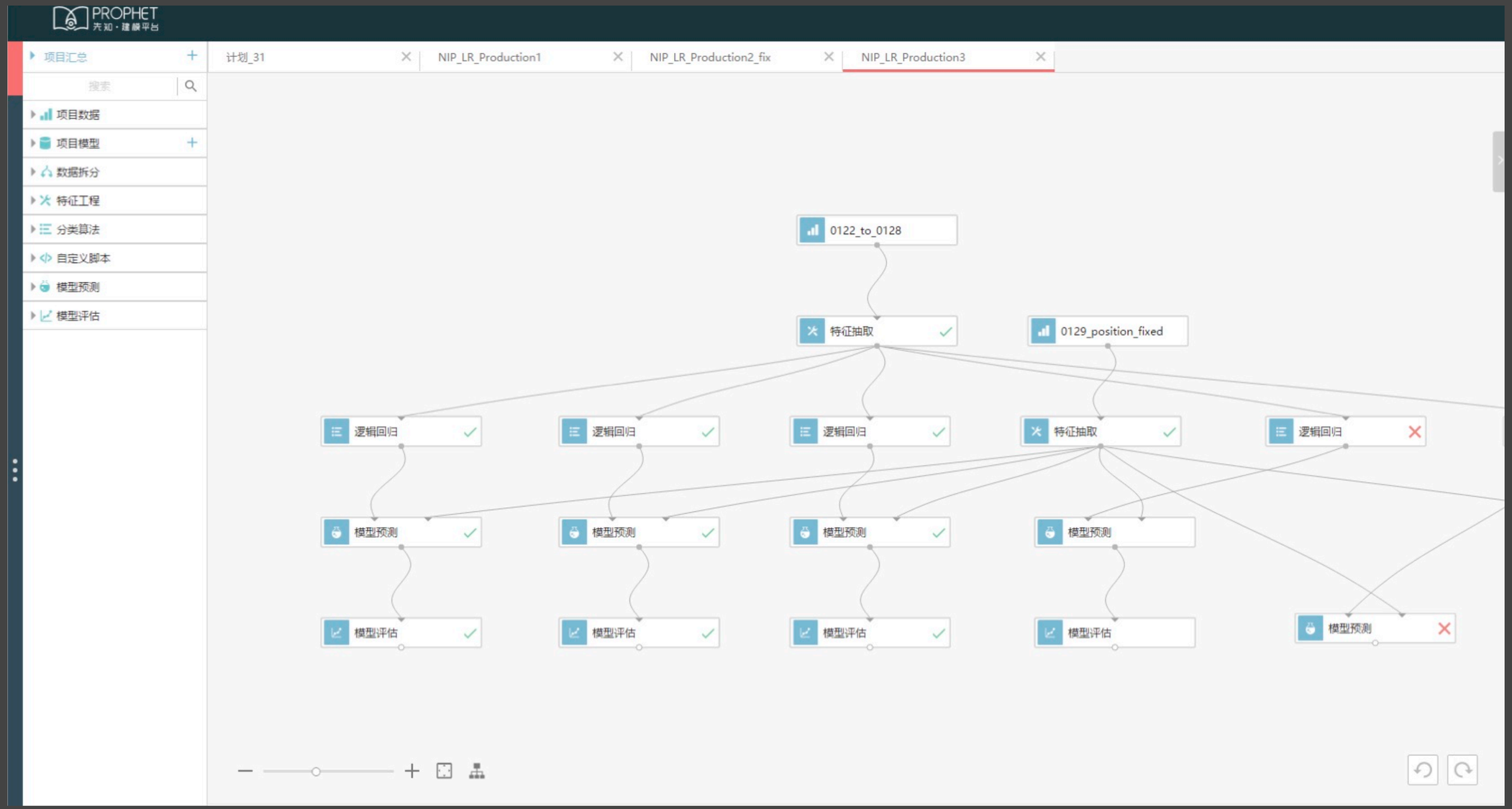
```
deployment_object = {
  "apiVersion": "extensions/v1beta1",
  "kind": "Deployment",
  "spec": {
    "replicas": 1,
    "template": {
      "spec": {
        "containers": [
          {
            "name": kubernetes_resource_name,
            "image": docker_image,
            "command": container_run_commands,
            "resources": {
              "requests": {
                "cpu": limit_cpu,
                "memory": limit_memory,
              }
            }
          }
        ]
      }
    }
  }
}
```

Step 3: Submit to Kubernetes

云机器学习平台 - 工作流

- 定义数据引入 -> 特征抽取 -> 模型训练 -> 模型评估工作流
- 拖拽式（Drag-and-drop）操作方式
- 自定义功能算子

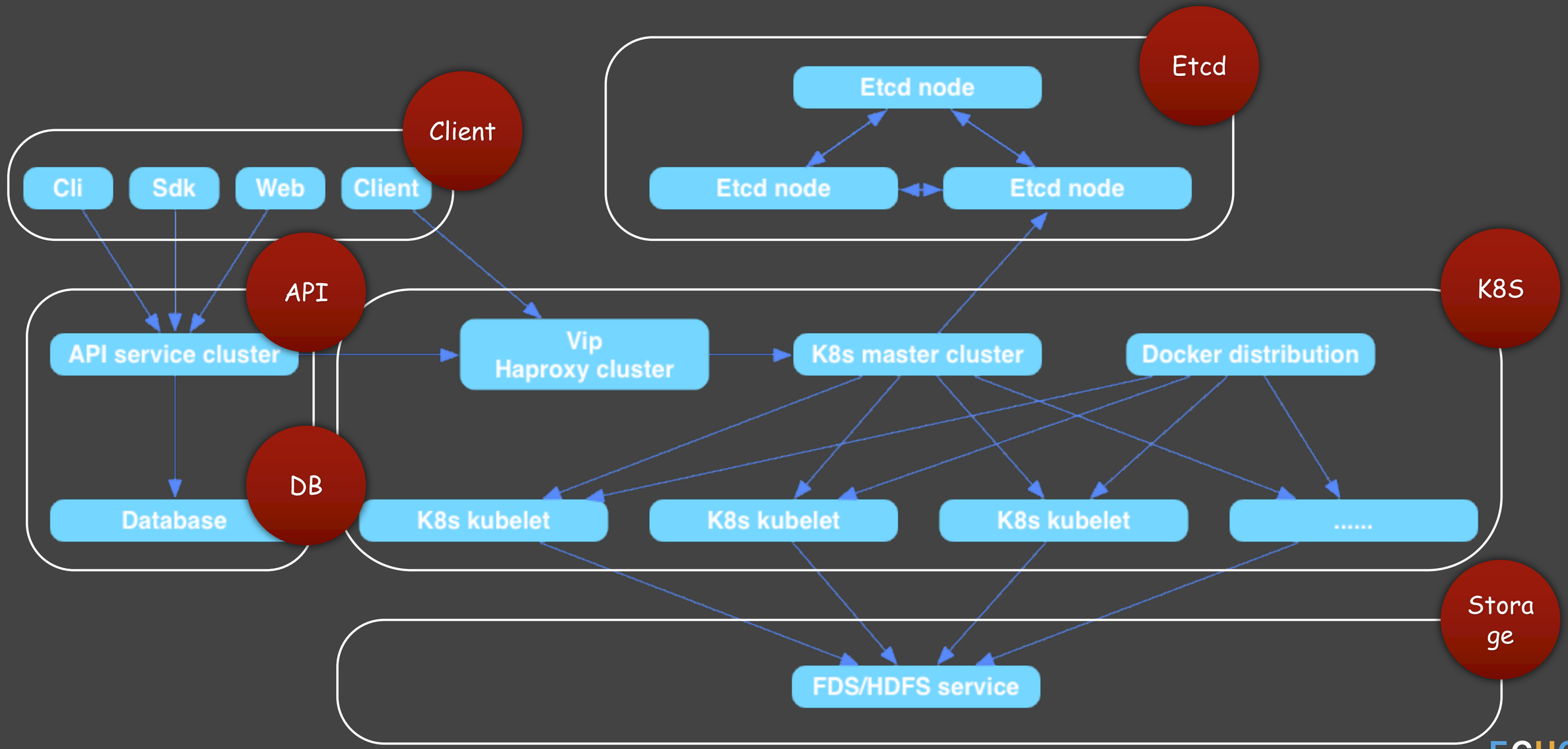
云机器学习平台 - 工作流


PROPHET
先知·建模平台
项目汇总
搜索
项目数据
项目模型
数据拆分
特征工程
分类算法
自定义脚本
模型预测
模型评估
计划_31
NIP_LR_Production1
NIP_LR_Production2_fix
NIP_LR_Production3
0122_to_0128
特征抽取
0129_position_fixed
逻辑回归
逻辑回归
逻辑回归
特征抽取
逻辑回归
模型预测
模型预测
模型预测
模型预测
模型预测
模型评估
模型评估
模型评估
模型评估

云机器学习平台 - 算子

- 最简抽象接口：abstract void execute()
- ~10 数据处理算子，10+ 机器学习训练算子，3 AutoML算子
- Hadoop/Spark/Kubernetes多集群支持
- More on https://www.4paradigm.com/support/help

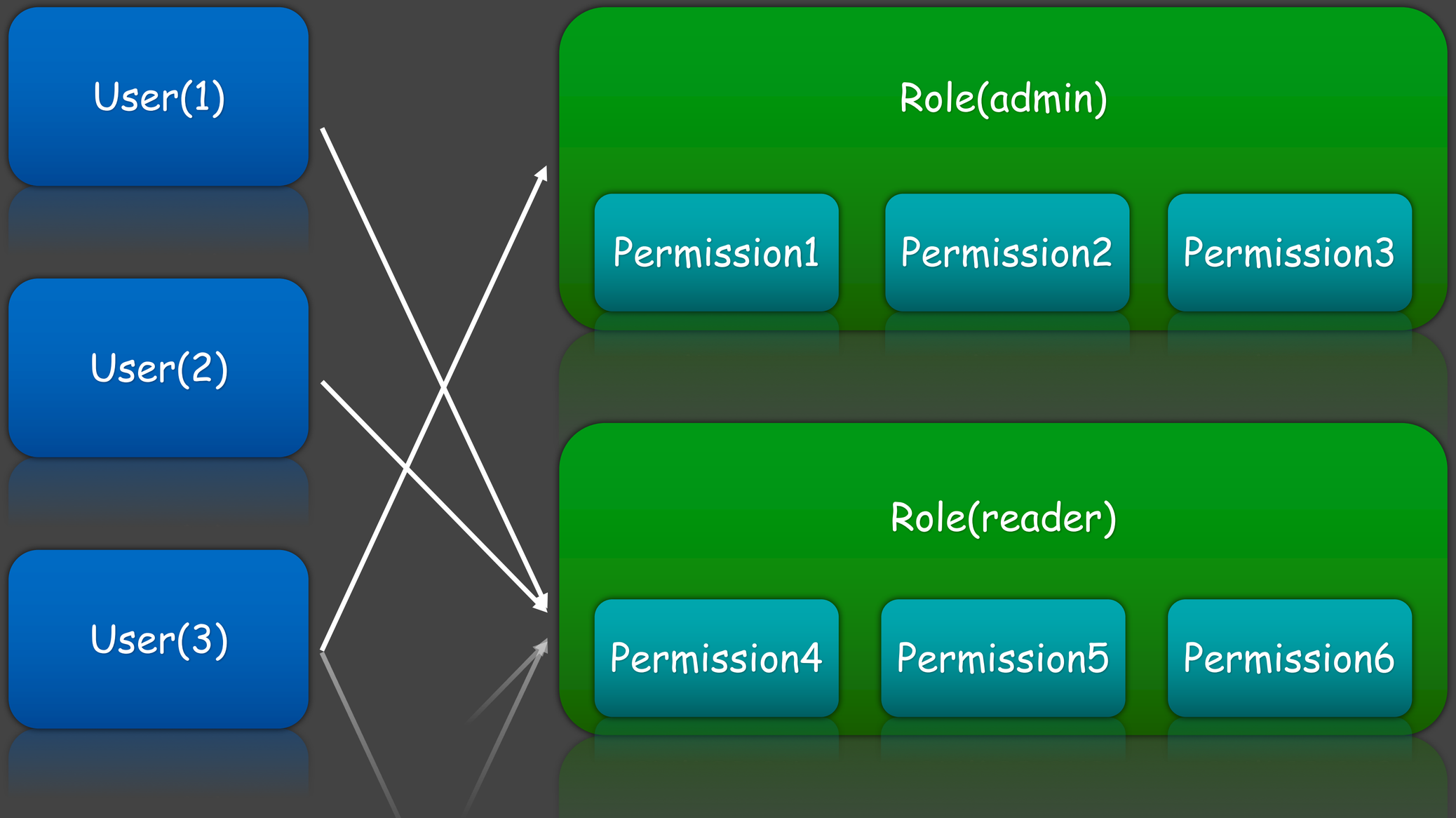
云机器学习平台 - 高可用


Client
Cli
Sdk
Web
Client
Etcd
Etcd node
Etcd node
Etcd node
API
API service cluster
Vip
Haproxy cluster
K8s master cluster
Docker distribution
K8S
DB
Database
K8s kubelet
K8s kubelet
K8s kubelet
.....
Stora
ge
FDS/HDFS service

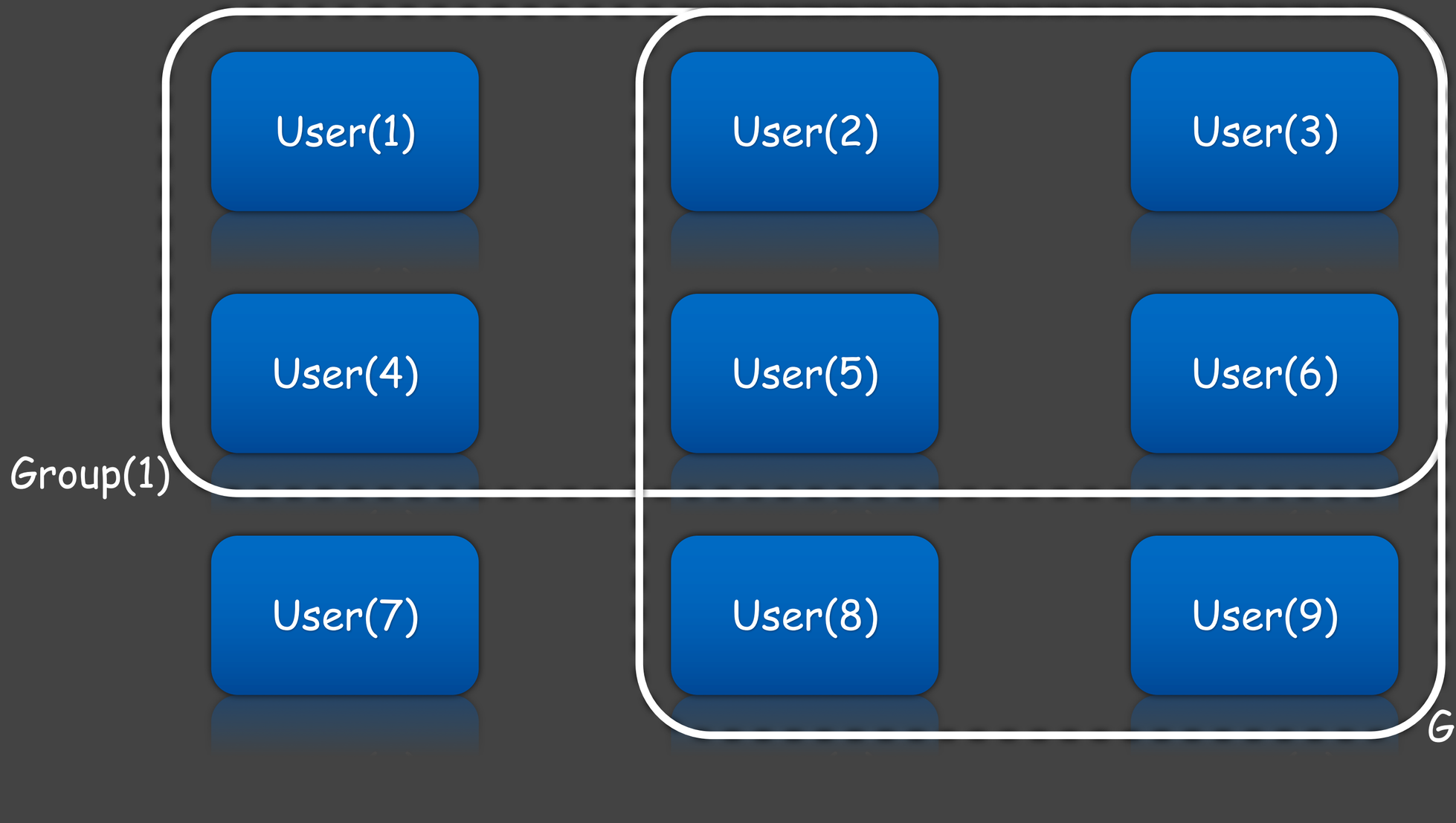
云机器学习平台 - 多租户

- Authentication
  - Username / Password
  - Access key / Secret key
  - LDAP
- Authorization
  - Role-based access control

云机器学习平台 - 多租户


User(1)
User(2)
User(3)
Role(admin)
Permission1
Permission2
Permission3
Role(reader)
Permission4
Permission5
Permission6

云机器学习平台 - 多租户


User(1)
User(2)
User(3)
User(4)
User(5)
User(6)
Group(1)
User(7)
User(8)
User(9)
Group(2)

Conclusion

- 人工智能与机器学习介绍
- 机器学习算法原理与实现
- 云机器学习平台架构实践

Thanks