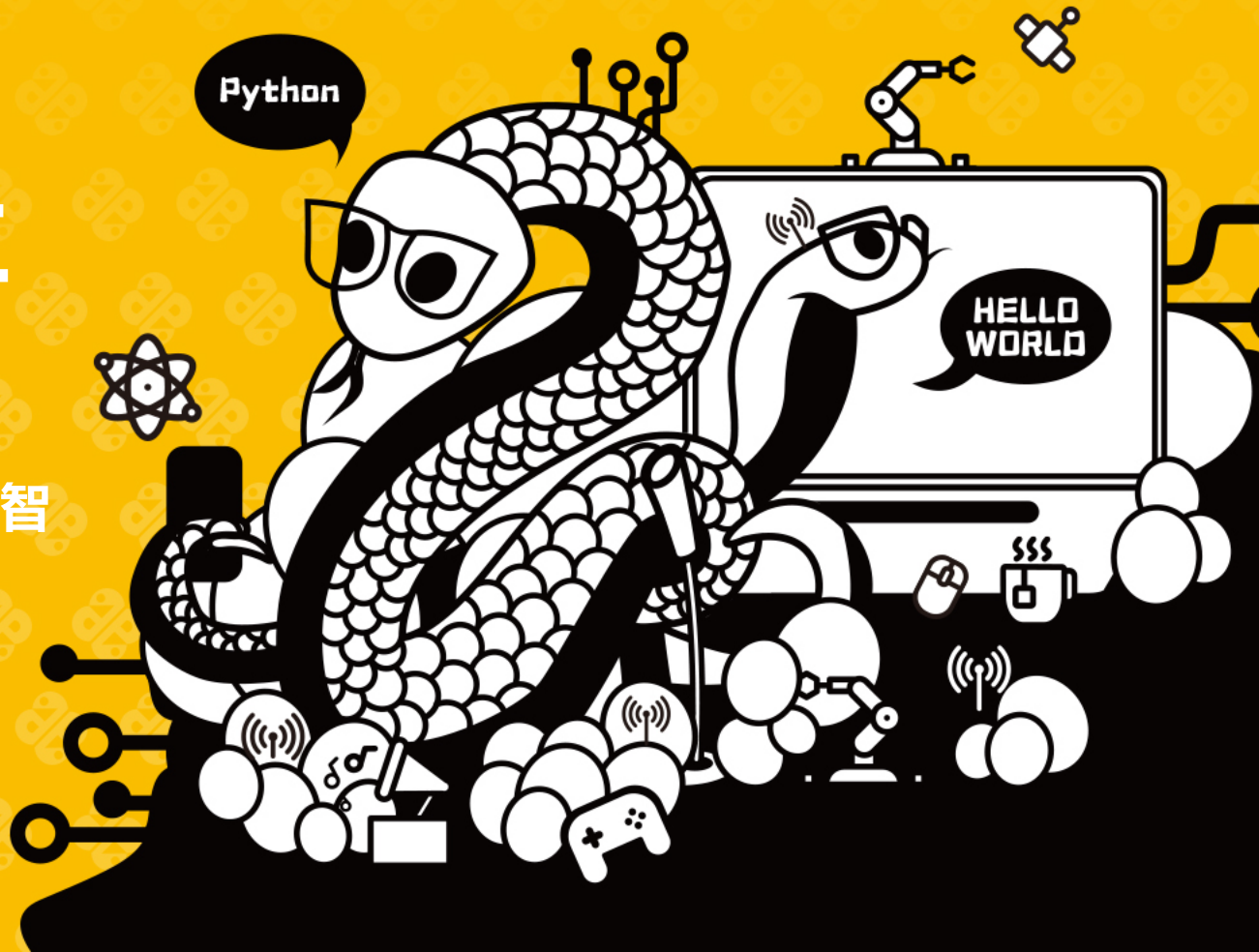
python × PYCHINA Python 中国社区

Python for Good

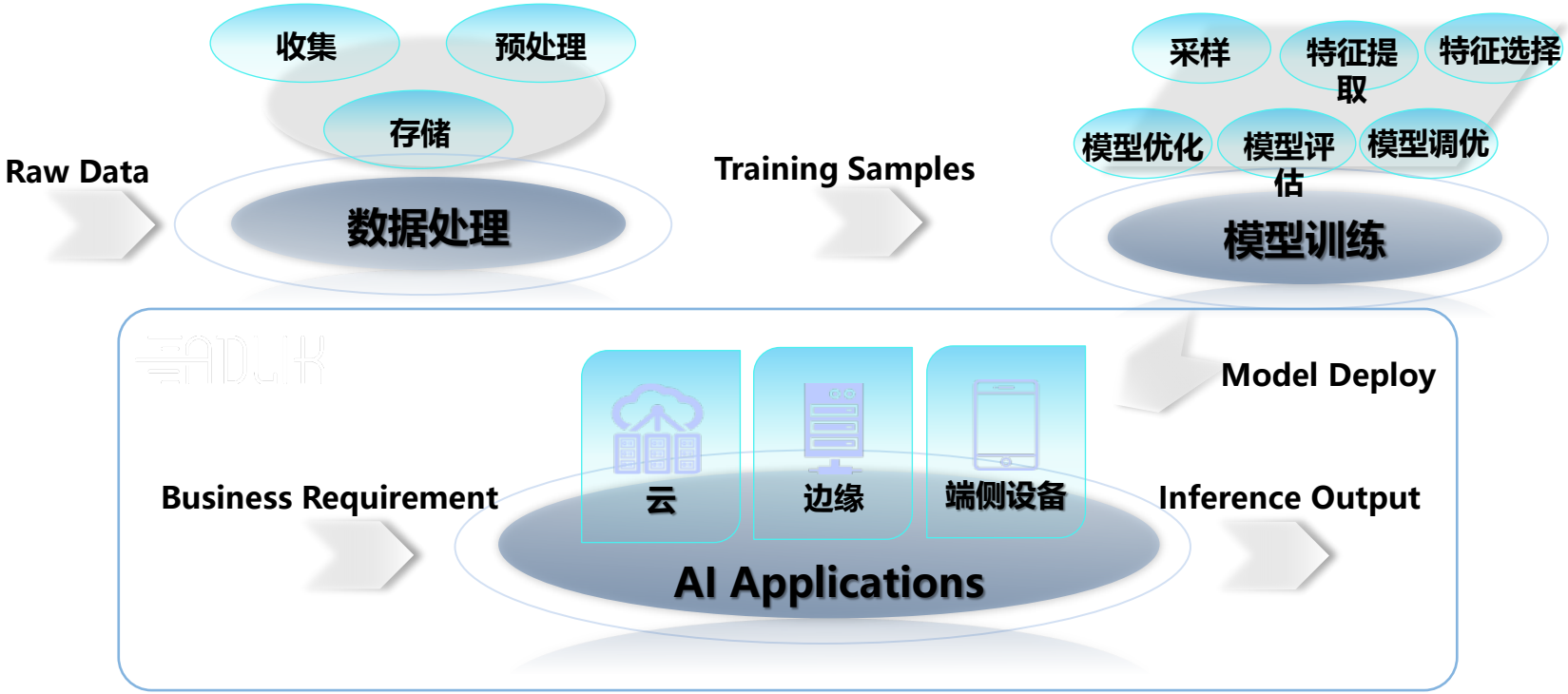
PyCon China 2022

Adlik深度学习推理加速工具链的应用

主讲人： 张凯莉 – 中兴通讯股份有限公司人工智能算法工程师

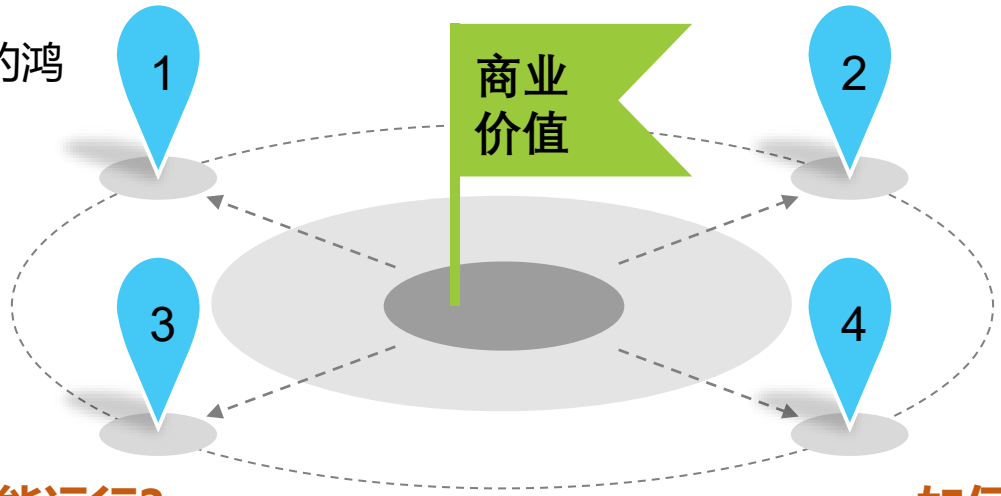


AI应用三个阶段


收集
预处理
存储
Raw Data
数据处理
Training Samples
采样
特征提取
特征选择
模型优化
模型评估
模型调优
模型训练
Model Deploy
Business Requirement
云
边缘
端侧设备
AI Applications
Inference Output

AI模型部署挑战

如何将模型和业务结合?
- 模型构建和模型使用之间的鸿沟
- 开发环境和生产环境的鸿沟



如何高效部署模型?
- 训练框架的差异
- 推理框架的差异
- 线上运行和线下训练的差异

如何确保模型以最优性能运行?
- 时延、吞吐量、内存占用
- 模型性能优化

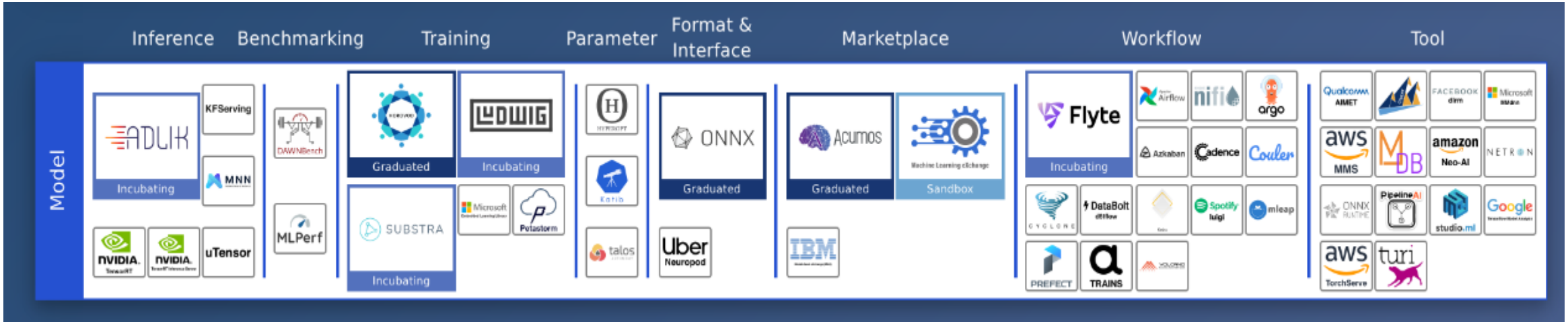
如何有效管理生产环境中的模型?
- 模型性能监控
- 模型版本管理
- 模型生命周期管理

AI模型部署对推理工具链提出的需求

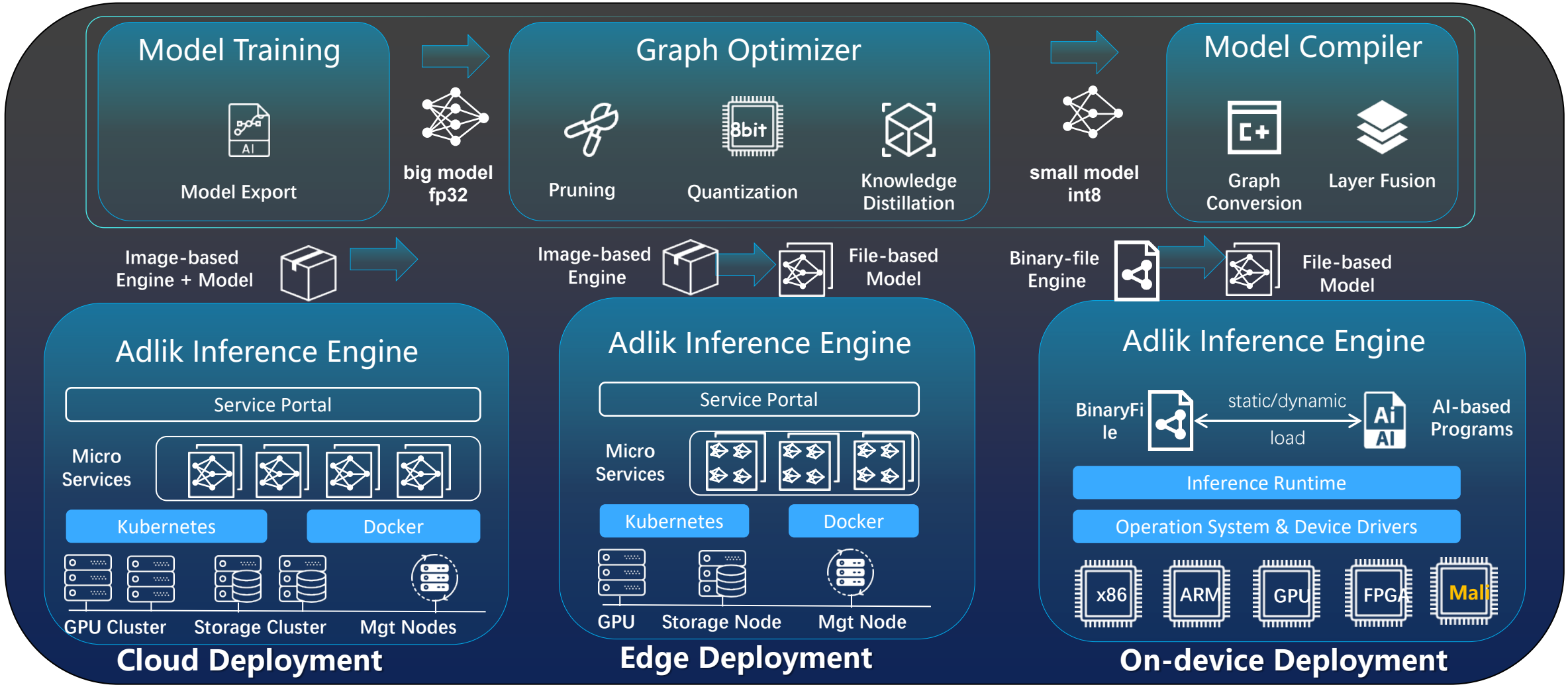
- 广泛支持主流训练框架
- 高效的模型压缩，优化能力
- 支持多种AI硬件
- 具备完善的工业应用特性，高可用、易运维
- 高性能，可以发挥硬件最佳性能
- 轻量化，可以满足边缘、终端各类部署环境
- 易用性，快速上手解决问题

Adlik: 深度学习推理加速工具链

- 一种将深度学习模型从训练完成，到部署到特定硬件并提供应用服务的端到端工具链，实现模型从研发状态到生产应用环境的高效切换
- 与多种推理引擎协作，提供灵活的模型加速、部署、推理方案，助力用户构建高性能AI应用
- 在Linux基金会AI和数据基金会（LF AI & Data）开源



Model Optimizer & Compiler:
提升模型计算效率，减少能耗，降低推理时延



Adlik Engine: 支持云边端等多种环境的模型部署

Adlik：解决模型部署过程中挑战性问题

更省

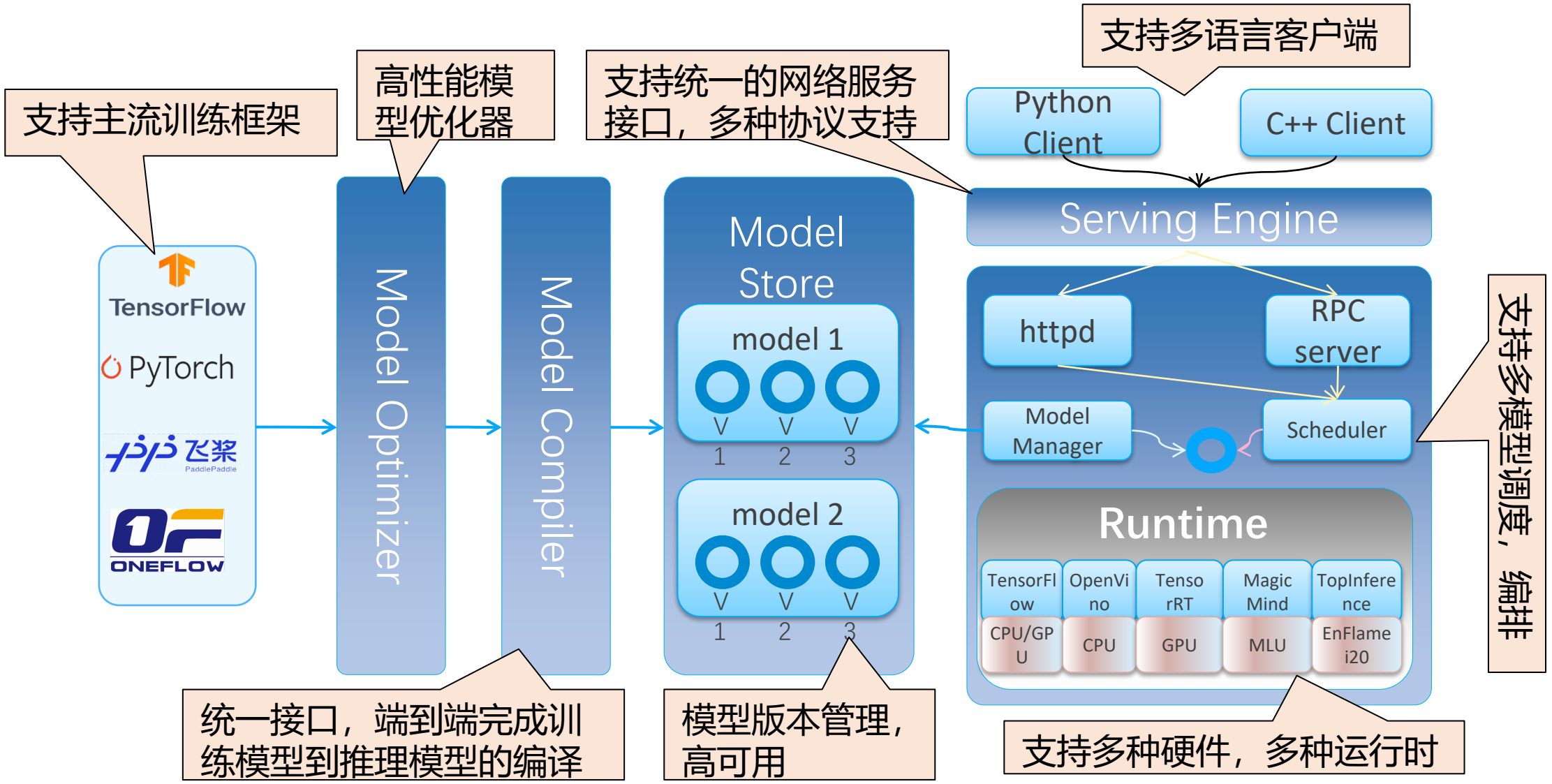
- 工程化参数自适应优化，降低部署复杂度，更省部署人力
- 简单方便的模型部署pipeline，缩短模型上线周期，更省部署时间
- 统一的模型推理和管理接口，更省模型迁移成本

更快

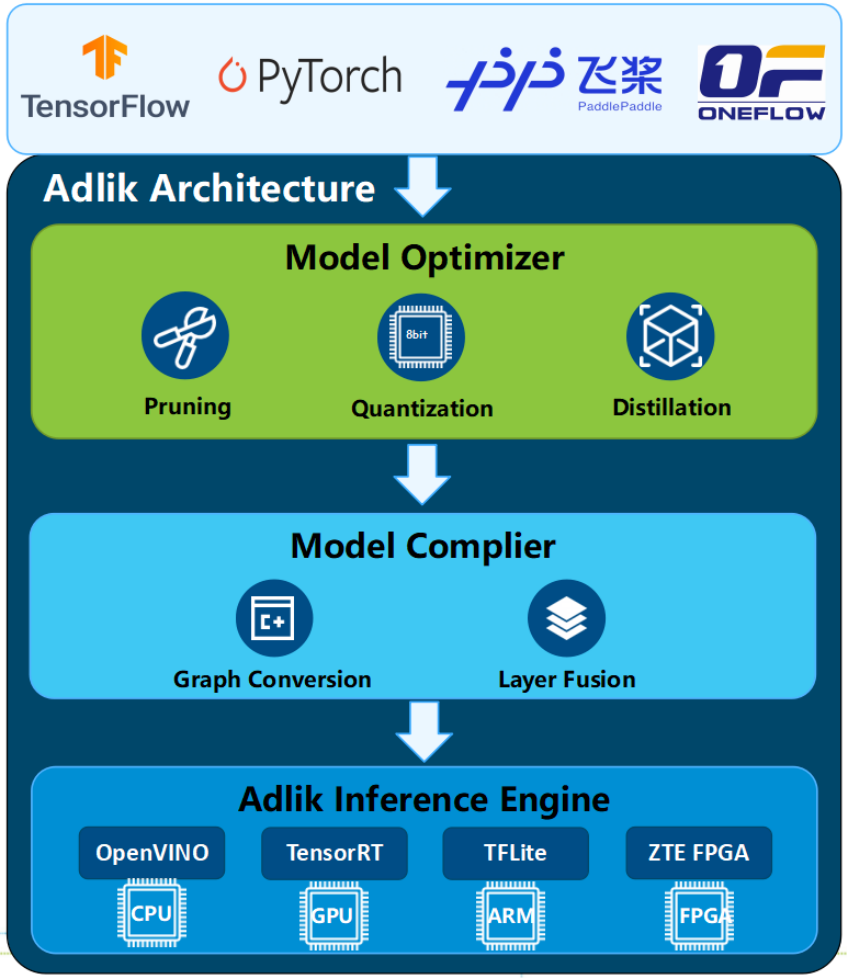
- 内置多种高性能运行时，以供用户更快地按需选用
- 提供可拓展性强的Serving SDK，可更快集成自定义推理运行时
- 提供灵活易用的推理API，更快实现AI应用的构建、迭代

更优

- 多种模型压缩、优化算法在实践中表现出出色性能
- 面对异构的部署硬件、不同的应用场景，提供更优的端到端方案，保障更优的推理性能
- 更优的模型管理、完善的模型运行监控


支持主流训练框架
高性能模型优化器
支持统一的网络服务接口，多种协议支持
支持多语言客户端
Python Client
C++ Client
Serving Engine
TensorFlow
PyTorch
飞桨 PaddlePaddle
ONEFLOW
Model Optimizer
Model Compiler
Model Store
model 1
V1 V2 V3
model 2
V1 V2 V3
httpd
RPC server
Model Manager
Scheduler
Runtime
TensorFlow
OpenVino
TensorRT
Magic Mind
TopInference
CPU/GPU
CPU
GPU
MLU
EnFlame i20
支持多模型调度，编排
统一接口，端到端完成训练模型到推理模型的编译
模型版本管理，高可用
支持多种硬件，多种运行时

Adlik模型优化器



- 支持多种结构化剪枝方法，并可通过多节点，多GPU进行剪枝；支持自动剪枝，根据网络类型（如ResNet-50 等）和限制条件（如FLOPs，Latency），就能自动决定模型每一层的通道数，得到在限制条件下最优的模型结构
- 支持多种蒸馏方法，使用剪枝和多教师蒸馏组合优化，进一步压缩模型，同时提升模型精度，完成图像分类，目标检测等任务精度提升
- 利用少量校准数据快速实现8-bit PTQ量化；支持QAT量化算法，提升量化模型精度

Adlik模型优化器：优化效果

Adlik 模型优化用于 ResNet50

- 使用自动剪枝工具剪枝，剪枝后吞吐量提升2.74倍
- 经过蒸馏，精度提升4.2%
- 使用INT8 PTQ量化，吞吐量再提升2.96倍，端到端加速13.82倍

模型优化方法	ThroughPut (OpenVINO)	Accuracy
baseline	432	76.80%
自动剪枝	1615	73.30%
自动剪枝+蒸馏	1615	77.50%
自动剪枝+蒸馏+INT8量化	6401	77.00%

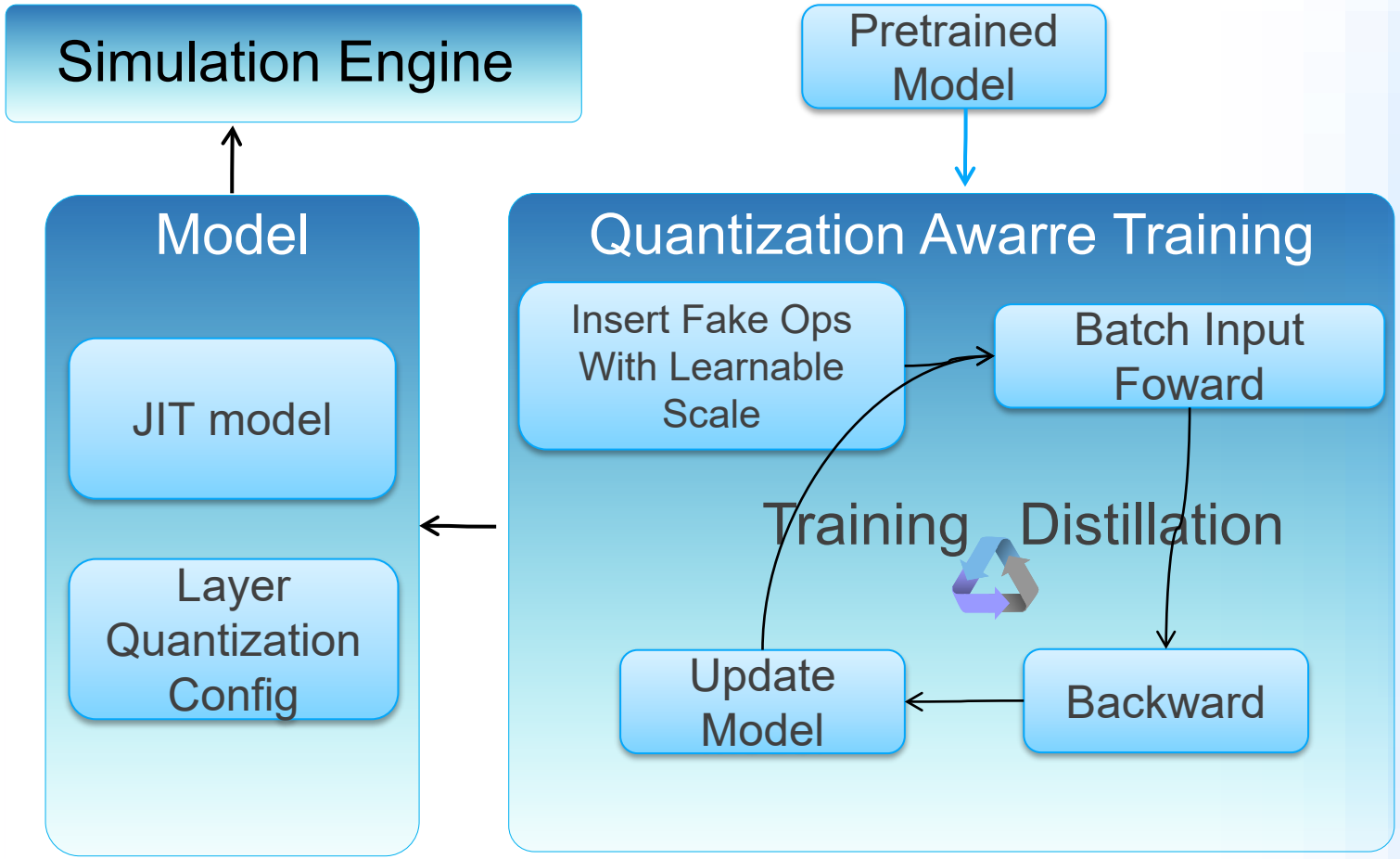
Adlik 模型优化用于 YOLOv5m

- 通过自动剪枝提升25%吞吐量
- 经过蒸馏，精度提升2.8%
- 通过INT8 PTQ量化，吞吐量再提升3.1倍，端到端加速3.33倍

模型优化方法	ThroughPut (OpenVINO)	Accuracy
baseline	73	44.70%
自动剪枝	95	41.80%
自动剪枝+蒸馏	95	44.60%
自动剪枝+蒸馏+INT8量化	316	43.90%

Test environment: Intel® Xeon® Platinum 8378C CPU @ 2.80GHz * 2

Adlik模型优化器： INT4量化框架



- 支持INT4模型量化
- 支持INT4全整形模型模拟推理
- 支持指定层不量化或者INT8等混合量化
- 支持导出为JIT文件
- 支持量化和蒸馏结合提升精度

Adlik模型优化器：LSQ量化算法

使用传统QAT量化，量化参数都是根据每一轮的权重/激活计算出来的，整个网络在训练的过程中只会更新权重的数值。而LSQ算法则把量化的scale也作为一个可以学习的参数（对称量化，zero_point为0），也就是说，每次反向传播的时候，需要对其求导进行更新。求导公式如右所示：

$$\bar{v} = round\left(clip(v/s, -Q_N, Q_P)\right)$$

$$\hat{v} = \bar{v} \times s$$

$$\frac{\partial \hat{v}}{\partial s} = \begin{cases} -Q_N & v/s <= -Q_N \\ -\frac{v}{s} + round(v/s) & -Q_N < v/s < Q_P \\ Q_P & v/s >= Q_P \end{cases}$$

其中，v是 float 的输入，$\bar{v}$ 是量化后的数据（仍然使用 float 来存储，但数值由于做了 round 操作，因此是整数），$\hat{v}$ 是反量化的结果。$-Q_N$ 和 Q_P 分别是量化数值的最小值和最大值（在对称量化中，Q_N、Q_P通常是相等的），s是量化参数。

LSQ： Learnable Scale Quantization， https://arxiv.org/abs/1902.08153

Adlik模型优化器：INT4量化效果

ResNet50 INT4传统QAT量化

- 使用传统QAT量化，INT4全4bit，精度67.182%
- 使用传统QAT量化，INT4量化（第一层和最后一层8bit）精度72.898%

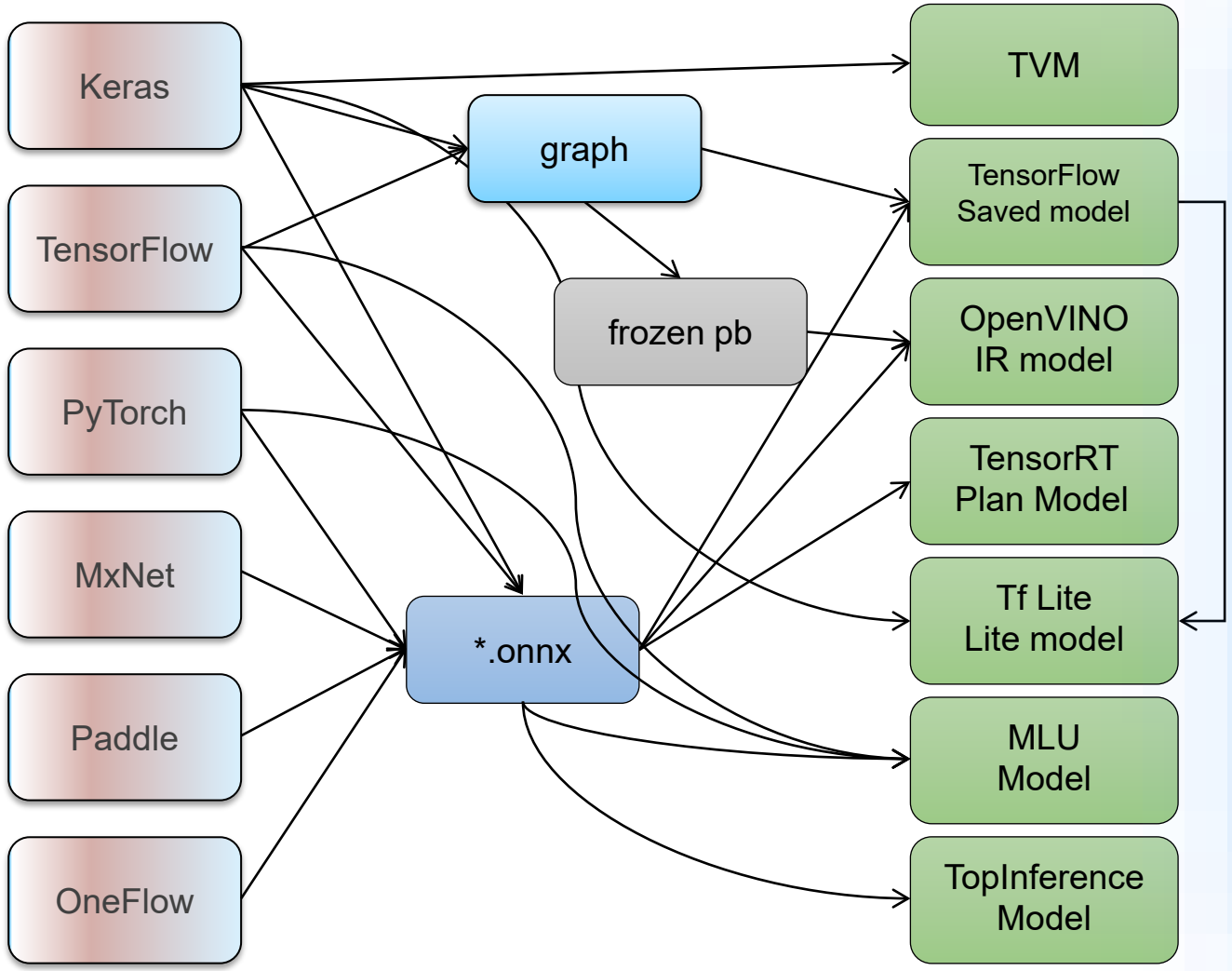
传统QAT量化方法	Accuracy
baseline	76.962%
int4量化	67.182%
int4量化+第一层和最后一层int8	72.898%

ResNet50 INT4 LSQ QAT量化

- 使用LSQ算法，INT4全4bit，精度能达到74.452%
- 使用LSQ算法，INT4量化精度能达到76.22%
- 经过蒸馏+量化，精度提升到77.878%

模型优化方法	Accuracy
baseline	76.962%
int4量化	74.452%
int4量化+第一层和最后一层int8	76.22%
蒸馏 + int4 量化	77.878%

Adlik编译器：支持多种训练框架接入



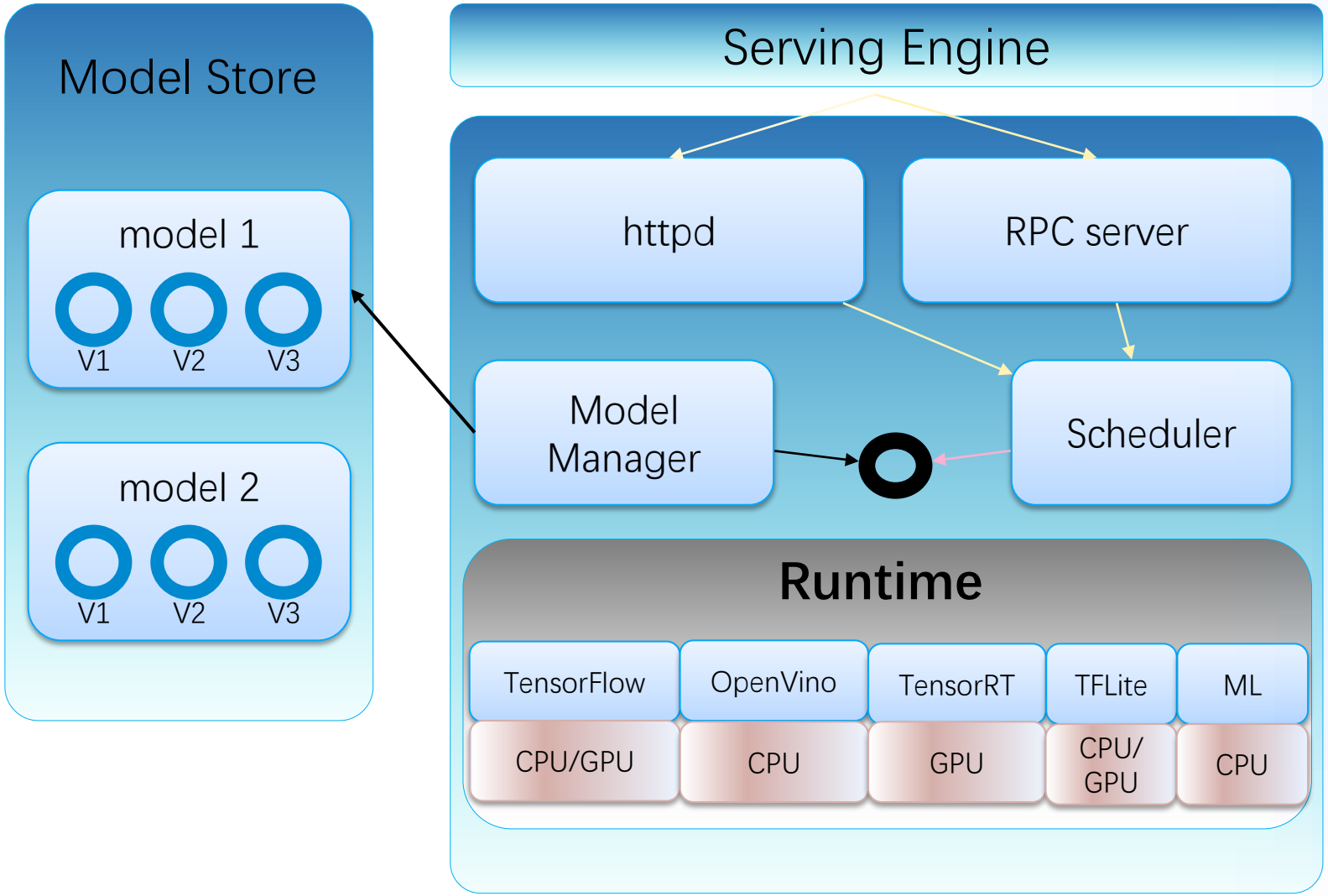
支持使用统一接口方案将多种原始训练模型格式转换到目标运行时模型格式

支持构建DAG 完成端到端的不同模型表达格式的转换

支持TfLite, TensorRT, OpenVINO模型量化

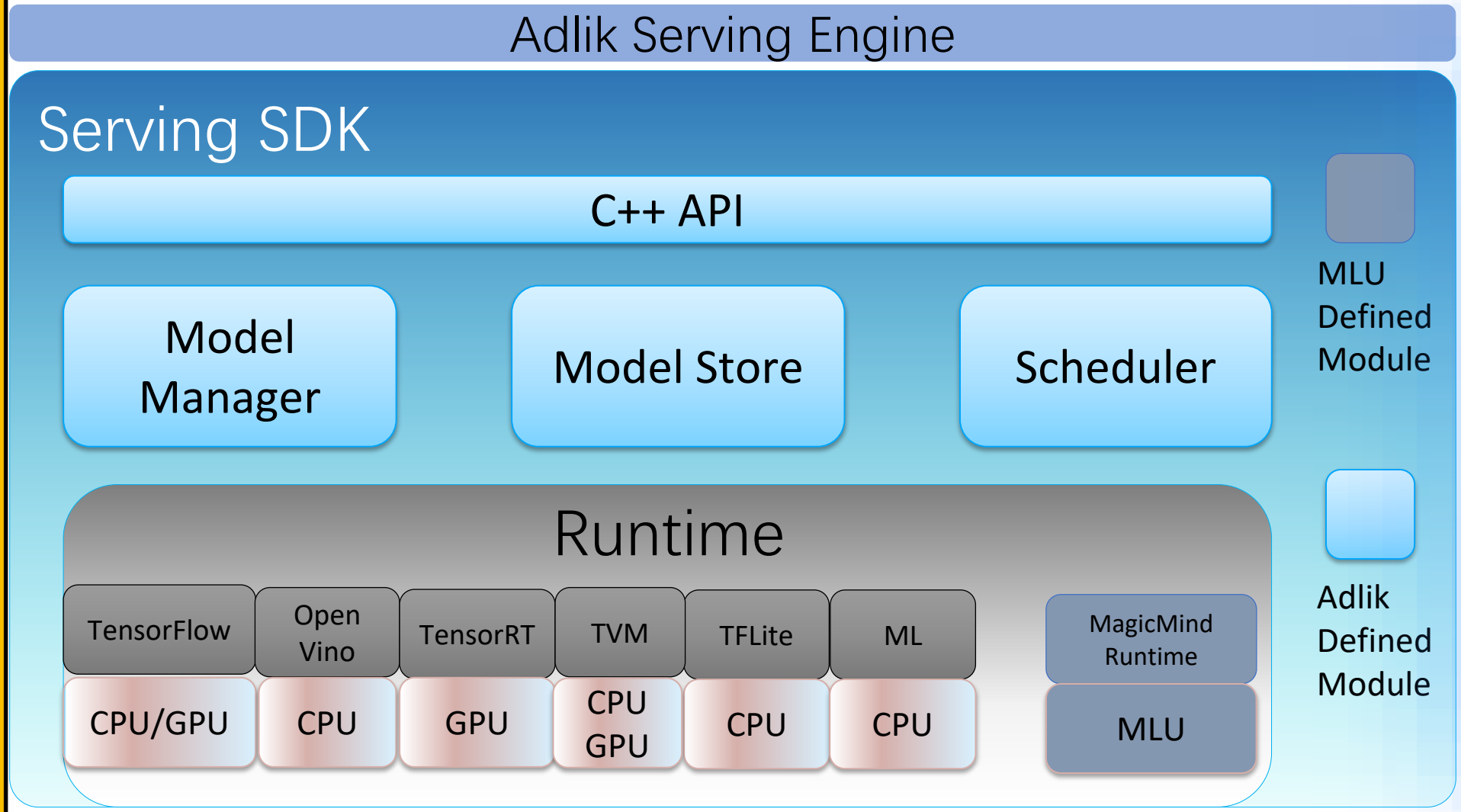
支持Paddle，OneFlow等国产训练框架，寒武纪，燧原等国产AI芯片

Adlik Serving Engine



- 支持模型上载，升级，版本管理，监控等功能
- 使用统一推理接口完成各类运行时的推理
- 多模型，多模型实例和多运行时多统一调度管理
- 支持用户自定义运行时
- 支持机器学习算法

Adlik Serving SDK：扩展新运行时和新硬件



C++ API

支持用户自定义运行时

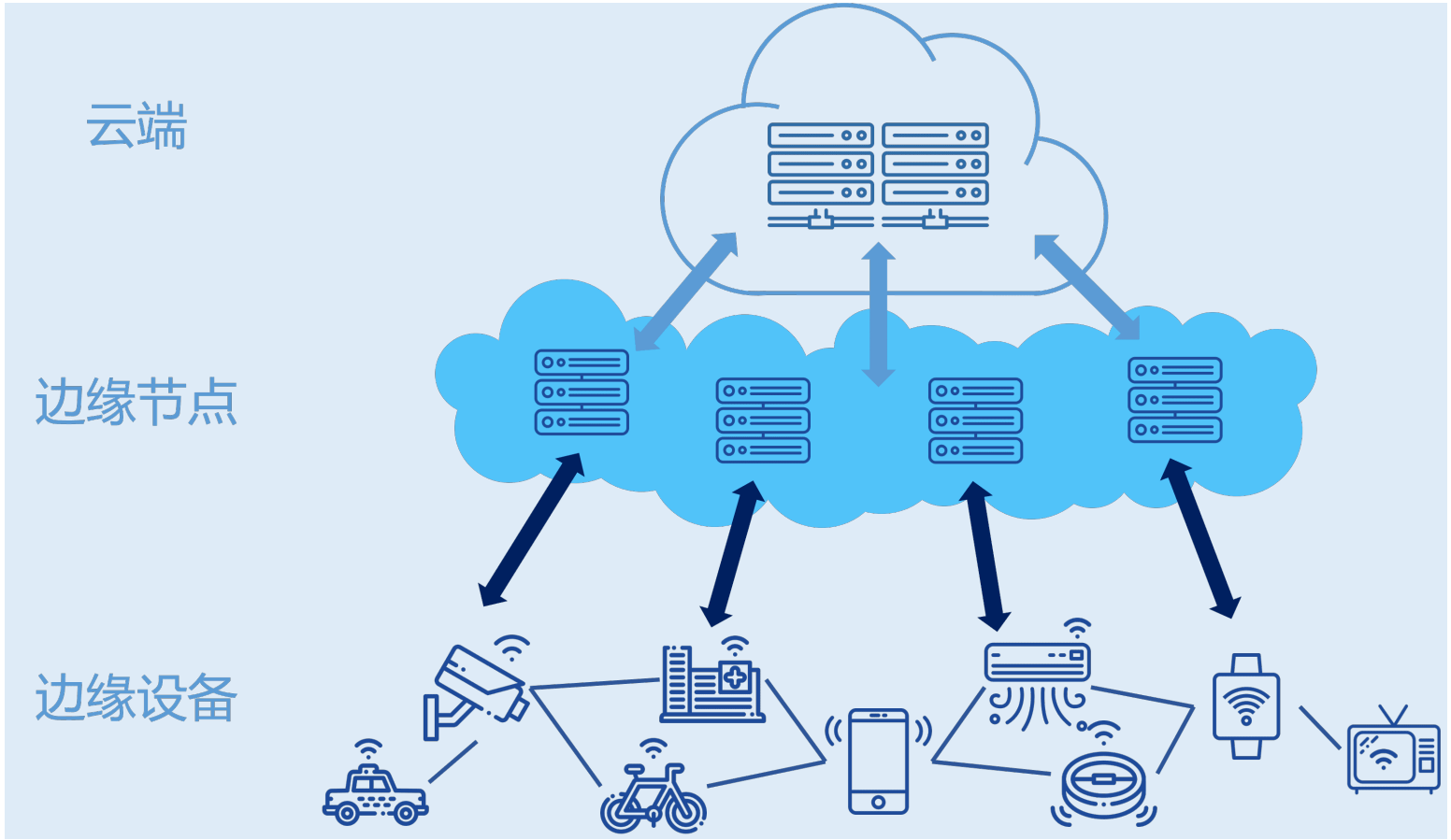
支持用户自定义Op

支持模型编排

用户易于扩展自己运行时和硬件

Adlik使用场景

- CV模型
- LSTM模型
- NLP模型
- 深度推荐模型



Adlik Demo

1. 从阿里云获取Adlik镜像

```
docker pull docker_image_name:tag
```

2. 运行模型编译器镜像，根据自定义的配置文件实现模型的编译

```
docker run -it --rm  -v source_model:/mnt/model
registry.cn-beijing.aliyuncs.com/adlik/model-compiler:v0.5.0_trt7.2.1.6_cuda11.0 bash
```

```
python3 "-c" "import json; import model_compiler as compiler;
file=open('/mnt/model/serving_model.json','r');
request = json.load(file);compiler.compile_model(request); file.close()"
```

3. 运行推理引擎，开始模型推理

```
docker run -it --rm  -p 8500:8500 -v compiled_model:/model
registry.cn-beijing.aliyuncs.com/adlik/seving_openvino:v0.5.0 bash
```

```
adlik-serving --grpc_port=8500 --http_port=8501 --model_base_path=/model
```

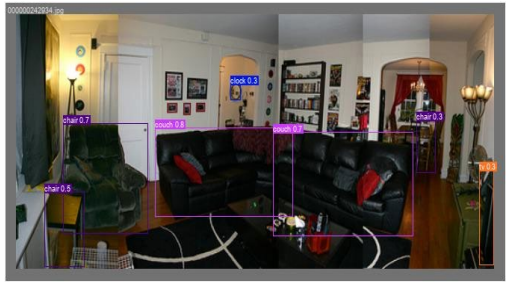
Adlik云原生运行Demo

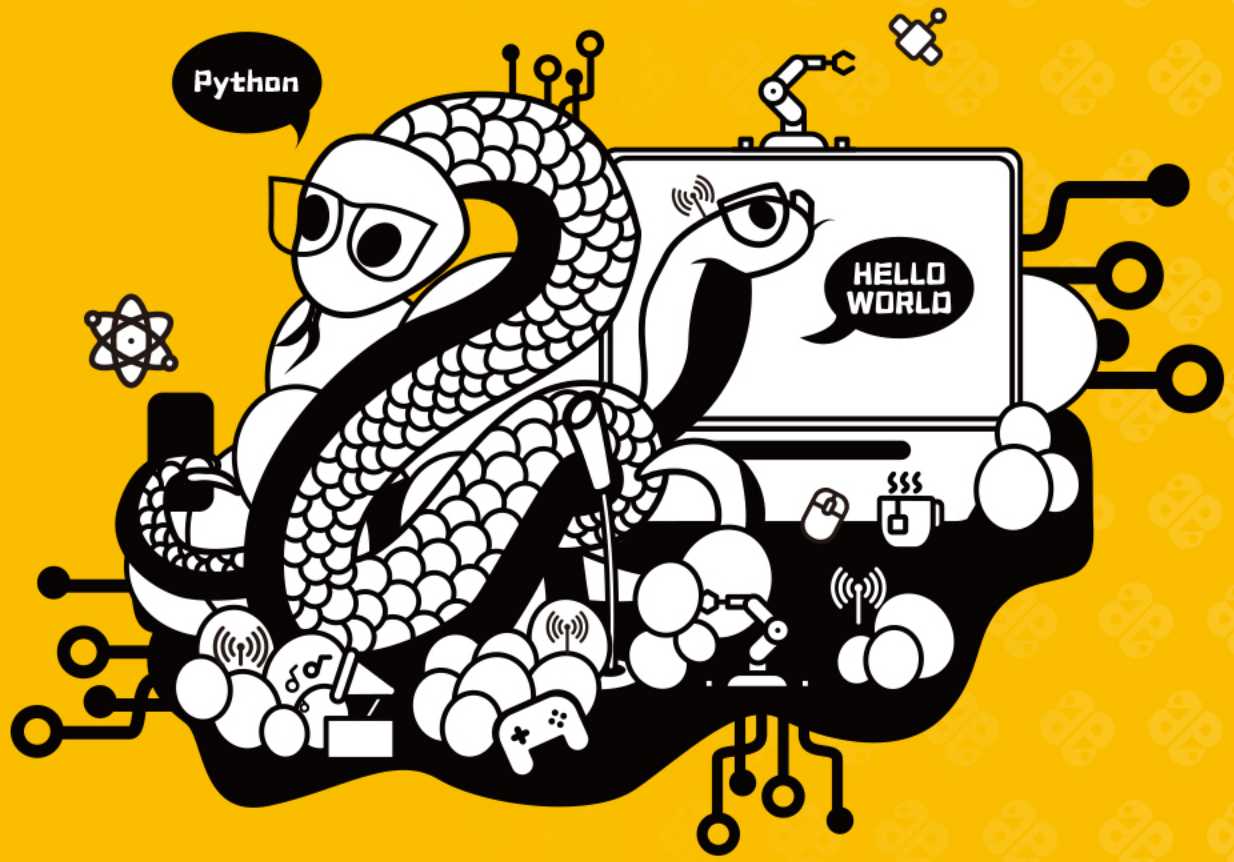
Docker Environment

```
docker run -it --rm -v /media/B/work/keras:/model 10.233.170.2:5000/adlik/model-compiler:v0.5.0_trt7.2.1.6_cuda11.0 bash
root@ecaf2fd16421:/# cd model/
root@ecaf2fd16421:/model# python3 compile model.py
Source type: ONNXModelFile.
Target type: OpenvinoModel.
Compile path: ONNXModelFile -> OpenvinoModel.
{'status': 'success', 'path': 'model_tf_yolov5_608_128/yolov5_1.zip'}
docker run -it --rm -v /home/t630/zkl:/model -p 31000:8500 10.233.170.2:31000/00253486/adlik_serving-openvino:v0.5.0 bash
/# adlik-serving --model_base_path=/model/yolov5_repos/ --grpc_port=8500 --http_port=8501
I adlik_serving/server/core/server_core.cc:54] Adlik serving is running...
I adlik_serving/server/grpc/grpc_options.cc:88] grpc server port: 8500
I adlik_serving/server/grpc/grpc_server.cc:24] grpc server is serving...
I adlik_serving/server/http/http_options.cc:35] http server port: 8501
python3 yolov5_client.py -n yolov5s -b 1 dog.jpg
```

Kubernetes Environment

```
kubectl create -f compiler.yaml
pod/model-compiler created
kubectl get pod | grep compiler
model-compiler                          1/1     Running           0          24s
ls
yolov5    yolov5_1.zip
kubectl create -f openvino-serving.yaml
kubectl get pod | grep openvino-serving
openvino-serving                        1/1     Running           0          24s
kubectl create -f openvino-svc.yaml
kubectl get pod | grep openvino-serving
openvino-service          NodePort    10.254.255.197   <none>          8500:31501/TCP                79s
python3 yolov5_client.py -b 1 dog.jpg
```





Thanks!

感谢观看