Solar Corona (Edge AI) Intro

we need a better name than solar corona, which has not been a good name since this Jan....
why AI@Edge?

• Data Privacy
• Lower latency than cloud
• large volumes data on edge, expensive to transfer to cloud, and unnecessary sometimes due to temporal and spatial locality.
Challenges of AI@Edge

- few-shot samples per edge for training, cold booting, hard to converge
- geo-distributed dataset across edges
- Non-I.I.D data across edges, the performance of universal AI model degraded on edge
- resource constrained on edge
what we proposed

• **It IS:**
  • an edge-cloud collaborative ML service based on KubeEdge, including a server on cloud, an agent at the edge, workers to run inferencing or training and a lib for interoperability with existing ML frameworks,
  • some edge-cloud collaborative ML features, including: joint inferencing, incremental learning and collaborative training (aka federated learning)
  • to help *Domain-specific AI Developers* to build and publish edge-cloud collaborative AI services/functions easily
  • to help *Application Developers* to use edge-cloud collaborative AI capabilities.

• **It’s NOT:**
  • to re-invent existing ML framework, i.e., tensorflow, pytorch, mindspore, etc.
  • to re-invent existing edge platform, i.e., kubeedge, etc.
  • to offer domain/application-specific algorithms, i.e., facial recognition, text classification, etc.
Service Architecture

- **Server on Cloud:**
  1. uniportal of EdgeAI,
  2. across-edges coordination

- **Agent @ Edge:**
  1. local controller
  2. manage local dataset and models

- **Workers:**
  1. do inferencing or training, based on existing ML framework;
  2. launch on demand, imagine they are docker containers;
  3. different workers for different features;
  4. could run on edge or cloud.

- **Lib:**
  1. expose the Edge AI features to applications, i.e. training or inferencing programs.

---

**EdgeAI Workers: Applications (Inferencing)**
- Applications: e.g. Facial Recognition, ASR, etc
  - MindSpore / TensorFlow / Pytorch
  - EdgeAI Lib
  - ML Models

**EdgeAI Agent@Edge**
- Incremental learning
- Joint Inference
- Collaborative Training
- Dataset Mngt/Labeling
- Model Mngt
- Peers Mngt/States Sync
- Monitoring
- Batch Job Mngt
- Resource Mngt

**EdgeAI Workers: Local Training**
- Training Programs: Build-in Training Algo, 3rd-party Training Algo
  - MindSpore / TensorFlow / Pytorch
  - EdgeAI Lib

**EdgeAI Server@Cloud**
- Edge AI Service API
  - Cross-Edges Collaborative Training
  - 3rd-Party Training Algo Mngt
  - Incremental learning
  - Joint-Models Building
  - Dataset mngt/Labeling
  - Model Mngt
  - Resource Mngt

**EdgeAI Workers: Cloud Training/Aggregation**
- Training Programs: Build-in Training Algo, 3rd-party Training Algo
  - MindSpore / TensorFlow / Pytorch
  - EdgeAI Lib

**KubeEdge@Edge**
- Atlas, Hilens摄像头, x86服务器 (GPU, D芯片)

**KubeEdge@Cloud**
- ML Services@Cloud: e.g. ModelArts

**EdgeAI Workers: Inferencing Backend@Cloud**
- Inferencing API
  - MindSpore / TensorFlow / Pytorch
  - EdgeAI Lib
  - ML Models

**EdgeAI Workers: Cloud Training/Aggregation**
- Training Programs: Build-in Training Algo, 3rd-party Training Algo
  - MindSpore / TensorFlow / Pytorch
  - EdgeAI Lib

**IaaS/PaaS@Cloud:** e.g. CCE/ECS

---

**Service Architecture**

- 1. uniportal of EdgeAI,
- 2. across-edges coordination

- **Agent @ Edge:**
  1. local controller
  2. manage local dataset and models

- **Workers:**
  1. do inferencing or training, based on existing ML framework;
  2. launch on demand, imagine they are docker containers;
  3. different workers for different features;
  4. could run on edge or cloud.

- **Lib:**
  1. expose the Edge AI features to applications, i.e. training or inferencing programs.
Service API and Lib interfaces (under discussion)

- Still under discussion
- 2 examples for incremental learning and Federated learning, which could be significant changed at the end
Feature: joint inference

1. 预训练: 用户开发者上传样本数据
   1. 迁移学习: 样本选择从公开数据集发现特征近似样本，准备训练数据。
   2. 多优化目标模型训练: 训练边云协同大小模型。边云模型优化目标不同，分别关注 recall 和 precision。

2. 边云协同推理: 边云大小模型协同推理。上云判断 Algo 决策将置信度低送到云上推理。
Feature: incremental learning

3A. Edge cloud协同增量学习(云端训练): 用户运维者云上标注; 云端增量训练小模型，完成更新后更新大小模型

3B. Edge cloud协同增量学习(边缘训练): 边缘标注，边缘增量训练; 边缘自动更细
Feature: collaborative training (fed learning)
Thanks
the workflow