KubeEdge BP API Documents

- API1: Kubernetes native APIs
- API2: KubeEdge APIs (Kubernetes API extensions)
- API3: ML inference framework APIs
- API4: ML management APIs
- API5: ML inference offloading APIs

The purpose of this document is to enumerate the APIs which are exposed by Akraino Blue print project to the external projects Akraino/Non Akraino for interaction/integration.

This document should be used in conjunction with the architecture document to understand APIs at a modular level and their interactions.

This document should function as a glossary of APIs with its functionality, interfaces, inputs, and expected outcomes as the following example:

API1: Kubernetes native APIs

API2: KubeEdge APIs (Kubernetes API extensions)

API3: ML inference framework APIs

API4: ML management APIs

This is the link of ML management API specification.

API5: ML inference offloading APIs

ML Offloading APIs provide synchronization of ML inference service with UE side. It serves application developers and enables machine learning apps to offload computation intensive jobs from UE device to close by edge nodes. ML offloading services satisfy the ML computing resource requirement, meanwhile its responses faster than cloud ML services.

The ML offloading APIs offer ML inference services (tensorflow serving frameworks) from KubeEdge sites, which contains a model pool. Pre-trained Machine Learning models can be deployed to the pool from cloud environment. In the future, the pool can open different categories of models to cover a wide variety of use cases in ML domain. if an app developers don't have a in-house trained model, they can also chose from the existing models, and it enables traditional app developers to quickly adopt the KubeEdge ML offloading solution without concerns of model management by themselves.

The KubeEdge ML offloading service has a Facial recognition demo api. The demo mobile application passes a face image via https request, and the edge ML offloading service identifies the expression and return corresponding facial expression code. Mobile app developers don't need to worry about the device resource limitation, or latency issues from the public cloud.

Here is an example of Facial expression API

Facial Expression Recognition

This operation takes an input image and success response will be in JSON format with 6 of human facial expression alone with different scores.

HTTP Method: POST

Request URL: https://{endpoint}/facialExpression

Parameters

Image type: PNG image

Image dimensions: greater than 48X48

Response

JSON:


```
[      "appID":"1234567",      "faceNumber":1,

 "emotion": {

   "anger": 0.0,

   "contempt": 0.0,

   "fear": 0.0,

   "happiness": 0.196,

   "sadness": 0.0,

   "surprise": 0.803

 }

]
```