New usecase - UAV Path Prediction

a. UAV Path Prediction (Guideline)

- **Introduction**
  - **We utilize collected data from unmanned aerial vehicle (UAV) movements in the x, y, and z axes as input for model training. The Long Short-Term Memory (LSTM) architecture is employed to train the model, and the OSC AIMLFW framework is used as the training model framework. Through this process, a model capable of predicting the path of UAVs is trained, providing a solution for various use cases.**
  - **Ensure that the Influx database is operational for this use case, which is based on the OSC AI/ML Framework (Release-H).**

- **Getting Started (Start from data insertion)**
  - Step 1. Query influx token

```
# Search the influx "token" value in below output
cat bitnami/influxdb/influxd.bolt | tr -cd "[:print:]"
```

  - Step 2. Create “UAVData” bucket (Inside Influx DB container)

```
influx bucket create -n UAVData -o primary -t <token>
```

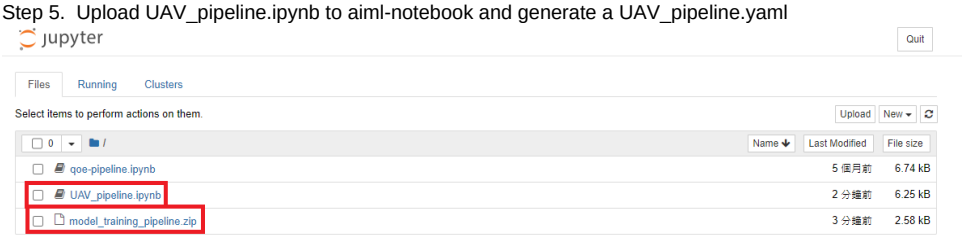
  - Step 3. Fill the file config (UAV_insert.py)

```
DATASET_PATH = '/path/to/dataset.csv' # Replace to the UAV dataset path
INFLUX_IP = 'localhost'               # Influx IP
INFLUX_TOKEN = 'VJpoNpqeVnjzvhpPm8jZ' # Influx token
```

  - Step 4. Excute the insert processing to insert data into Influx DB

```
python3 insert.py
```

  - Step 5. Upload UAV_pipeline.ipynb to aiml-notebook and generate a UAV_pipeline.yaml



  - Step 6. Start a UAV model training job
    blocked URLblocked URL

- Step 7. Load UAV model

```yaml
apiVersion: "serving.kserve.io/v1beta1"
kind: "InferenceService"
metadata:
  name: uav-model
spec:
  predictor:
    tensorflow:
      storageUri: "http://localhost:32002/model/test1208/1/Model.zip"
      runtimeVersion: "2.5.1"
      resources:
        requests:
          cpu: 1
          memory: 0.5Gi
        limits:
          cpu: 2
          memory: 0.5Gi
```

```
kubectl apply -f uav.yaml -n kserve-test
```

```
root@mitlab-virtual-machine:/home/mitlab/osc/aimlfw-dep# kubectl apply -f uav.yaml -n kserve-test
inferenceservice.serving.kserve.io/uav-model configured
root@mitlab-virtual-machine:/home/mitlab/osc/aimlfw-dep# kubectl get pods -n kserve-test
NAME                                                      READY   STATUS    RESTARTS   AGE
qoe-model-predictor-default-00001-deployment-68d85bf59b-45j4g   2/2     Running   0          124d
uav-model-predictor-default-00001-deployment-69d4f54ddf-18nhd   1/2     Running   0          2s
```

- Step 8. Start model prediction
  1. obtain Ingress port for Kserve (kubectl get svc istio-ingressgateway -n istio-system)
  blocked URL
  2. source uav.sh

File List

- UAV_dataset.csv (Download)
  The file contains collected UAV movement path data.
- UAV_insert.py (Download)
  The file processes the UAV_dataset and inserts the data into InfluxDB.
  (Changed required: **DATASET_PATH , INFLUX_IP , INFLUX_TOKEN**)
- UAV_pipeline.ipynb (Download)
  The file defines the model structure and training process.
- UAV_deploy.yaml (Download)
  The yaml file is used for deploying model inference service.
- UAV_input.json (Download)
  The json file is the sample data for the prediction.
- UAV_predict.sh (Download)
  The script used for excuting the model prediction.

Example

- Input:
  This input data represents a collection of points in a three-dimensional space, with each point defined by a set of three coordinates corresponding to the x, y, and z axes (After normalization).
  - Example input

UAV_input

```
[
     [0.7453425167510083, 0.057188434013895506, 0.3883151002000583],
     [0.9348491377517286, 0.5797269721343681, 0.15480809966429032],
     [0.32685562293198867, 0.40325520693639316, 0.37923173514277997],
     [0.07019999536045685, 0.26259026009845376, 0.2890220776419392],
     [0.562592860552039, 0.471677885159695, 0.41299704269747306],
     [0.8156868208972661, 0.6577897226335571, 0.5894501267006358],
     [0.6789936054809089, 0.5219624142693665, 0.4196227577850472],
     [0.10702478626584111, 0.3030673792265356, 0.5265722005767963],
     [0.9762029048313448, 0.30233720269747666, 0.4147831180404825],
     [0.5277344636914802, 0.1058208829010896, 0.9369119762104259]
   ]
```

- Output:
  The output should be next xyz-axis path prediction (After normalization).

    - Example output

UAV_output

```
Inference Result: [[-0.07196578  0.05871227 -0.00614708]]
```

Contributor

- Joseph Thaliath - Samsung
- Antony Wang - NTUST MITab
- Jasmine Lee - NTUST MITLab