Integrating REANA into ROB

Ajay R. Rawat
IRIS-HEP fellow
University of Washington, Seattle
Reproducible Open Benchmarks for Data Analysis Platform (ROB)

- Controlled competition-style environment platform
- Evaluate data analysis workflows
- Benchmark different workflows and rank them
- Reproduce the results of a workflow
Inspiration for ROB

- Top Tagger comparison (https://arxiv.org/abs/1902.09914)

**Task**: Classify the Top Quark jets From the background QCD jets.

**Benchmarks**:
- AUC of ROC curve
- Accuracy
- (and Rejection power [not shown here])
ROB Goals

ROB is designed to expand the approach used in Top Tagger comparison and to generalise it for all data analysis workflows

**Goals:**

(1) reduce the amount of time required to organize and evaluate such benchmarks

(2) ensure reproducibility of benchmark results.
ROB:
3 layers

- User Interface (Command Line Client, Web User Interface)
- Flowserv
- Workflow Engine: (local or cloud like REANA/AWS)
Benchmarking

- Create a task and the dataset
- Users submit their workflow and code
- ROB runs the workflow on the specified engine
- Benchmarks such as model accuracy/ AUC of ROC curve are obtained using the testing data
Benchmarking (contd.)

- Users can provide environments like Docker to run their workflow
- All of the workflows are then ranked based on these benchmarks
- They are evaluated on the same dataset
What is REANA?

REANA is a reproducible analysis platform allowing scientists to run containerised data analysis pipelines on remote compute clouds.

Image from:
http://docs.reana.io
REANA workflows:

REANA workflows have mainly 3 sections:

- **Inputs:**
  - Input files
  - parameters

- **Workflow:**
  - Commands to run this workflow

- **Outputs (files) produced by the workflow**

---

**REANA hello world YAML file**

```yaml
version: 0.3.0
inputs:
  files:
    - code/helloworld.py
    - data/names.txt
parameters:
  helloworld: code/helloworld.py
  inputfile: data/names.txt
  outputfile: results/greetings.txt
  sleeptime: 0
workflow:
type: serial
specification:
  steps:
    - environment: 'python:2.7-slim'
      commands:
        - python "${helloworld}"
          --inputfile "${inputfile}"
          --outputfile "${outputfile}"
          --sleeptime ${sleeptime}
outputs:
  files:
    - results/greetings.txt
```
Using a REANA cluster

- Use an existing REANA cluster eg. at CERN
  - If you have a cern account you can get an access token

- Deploy a REANA cluster on your local machine/server
Running workflows on REANA

- Command Line Interface `reana-client`
- Python Package

Your REANA token

In order to use your token, make sure you have reana-client installed and run:

```
$ export REANA_SERVER_URL=https://reana.cern.ch
$ export REANA_ACCESS_TOKEN=[token]
```
Integrating REANA in ROB

For ROB to run workflows on REANA, it uses the REANA python package to:

- Create a workflow on REANA by converting the YAML file into JSON
- Uploading all the input files to REANA server
- Starting the Workflow
- Monitoring the status of the workflow at certain time intervals
- Downloading the results once the workflow has finished execution
Benefits of integrating REANA in ROB

- All the required information to reproduce a workflow is stored on the REANA server.
- REANA server also stores the logs and output files.
- ROB can now run computationally heavy workflows on specific REANA clusters.
- ROB can now run multiple submissions at a time on REANA and then rank benchmarks locally.
Sources

- https://github.com/scailfin/rob-client
- https://github.com/scailfin/rob-ui
- http://docs.reana.io