**End-To-End AI Benchmarking**

**Summary Report**

Scientific Machine Learning Research Group

Scientific Computing Department

Rutherford Appleton Laboratory

Science and Technology Facilities Council.

May 2021

|  |
| --- |
| 1. **Synopsis** |

Benchmarking of machine learning solutions to scientific problems is an important initiative within the Scientific Machine Learning community. Although machine learning models and algorithms play a vital role in AI-specific solutions, there are a number of other components that contribute to the overall performance of the workflow of the solution that are often not included. For instance, data transformations, data movements and even some compute-intensive operations which are not the focus of the machine learning component, often go unmeasured in recording the performance of machine learning-driven solutions.

The main scope of this E2E project, supported by a funding from Intel (through the Alan Turing Institute), was to develop robust techniques for understanding and benchmarking scientific workflows in an end-to-end manner. The phrase end-to-end is used to mean several processes executed in a pipelined fashion. In addition to developing these techniques, it will also be of interest to measure and record the end-to-end performance of several different scientific workflows on a variety of hardware platforms.

|  |
| --- |
| 1. **Scope & Objectives** |

The scope of the project was to build two end-to-end AI benchmarks from two different scientific domains, with each of them having multiple stages including a machine learning component. The objectives included:

1. Developing techniques for assessing the performance of individual stages of the workflow. Such techniques, particularly that are of focussed on scientific cases, would help Intel in utilising similar methods across different cases and understanding the importance of individual stages;
2. Benchmarking and reporting on the performance of these workflows for different hardware platforms. The detailed benchmarking exercise would highlight the strengths and weaknesses of different platforms across the workflow and within stages, such that it may help improving the hardware or software ecosystem performance for ML solutions from Intel.

|  |
| --- |
| 1. **Scientific Cases** |

The overall project relied on two end-to-end cases gathered from two scientific domains. Each of these cases contain (at least) one stage where machine learning algorithm plays a key role, and at least another stage with generic processing.

**Case I: Structure Identification in Cryo-EM Data**

One of the challenges in structural biology is accurately identifying the structure of macromolecular assemblies in atomic detail. In this context, electron cryogenic electron microscopy(Cryo-EM) is one of the principle techniques for the determination of structures of macromolecules. It is particularly suited for large assemblies and/or membrane proteins which are important pharmaceutical targets not easily amenable via other biophysics methods such x-ray crystallography or nuclear magnetic resonance spectroscopy.

The overall determination of the structure by Cryo-EM is not a single-step process. Instead, the process evolves through a number of connected, complex stages, including denoising images, identifying and classifying molecules in the recorded 2D images, determination of the orientation of molecules and averaging to reconstruct a 3D volume, to finally fitting atomic models into the experimentally derived 3D volumes. More importantly, the exact steps vary by the type of molecular structures or the datasets of interest. As such, identifying a fully uniform set of stages, particularly those that are automatic, is a challenging task.

This end-to-end workflow benchmark concerns only part of this processing pipeline, but one that is useful enough to produce very tangible outputs at every stage of the pipeline, and also as a whole.



**Figure 1: End-to-End Processing in Structure Identification in Cryo-EM Data**

Owing the complex nature of the pipeline stages here, it requires fair amount of intervention at times (or intermediate outputs). The exact stages are included case 1 reports (case1/reports in the relevant github link).

**Case II:** **Cloud Masking for Surface Temperature Estimation**

Satellite data have become an integral part of the remote sensing tasks, particularly that of deciding the surface characteristics of the Earth. One of these tasks is the estimation of land and sea surface temperature (LST/SST) by masking and filtering clouds from satellite imagery. The central idea is to use machine learning for classifying each pixel of satellite imageries as cloud or non-cloud, with associated uncertainty. Although there are a number of similar efforts across different satellite and sensor modalities, imageries from the Surface Land Sea Temperature Spectrometer (SLSTR) on board the Sentinel-3 satellites is gaining widespread acceptance. The efficacy of the classification and estimation can be validated using a match-up of classified pixels to readings from sea surface buoys, as well as a thorough comparison with a small number of hand-labelled pixels. This end-to-end case study will focus on performing the overall operation of surface temperature estimation from a set of inputs, covering the stages of

1. pre-processing (image resizing, normalization, saturation management, and generation of random crops of images),
2. Cloud classification (training & inferencing); and
3. Estimation of the sea surface temperature at the pixel level.

These stages are depicted in Figure 2 below.



**Figure 2: End-to-End Processing in Cloud Masking**

|  |
| --- |
| 1. **Benchmarking Scope** |

These two workflows, in their own, provided an avenue for testing and evaluating different aspects of the overall workflow, such as, overall capability of the hardware system under test, capability of the system for training on data, and capability of the system for inferencing. As part of this process, we collected all relevant performance data linked to the stages of the workflow. We carried out the benchmarking on two systems, namely V100-equipped PEARL system, and ICX equipped system in EPCC.

|  |
| --- |
| 1. **Deliverables** |

We delivered the following for each of the cases:

1. Relevant source code for cases I and II (available at the GitHub Link [here](https://github.com/stfc-sciml/e2e/))
2. Download script to get hold of relevant datasets from the object storage (specific details are included in the relevant case documentations). The datasets are stored inside a publicly accessible bucket (details are available inside the github).
3. Reports on the benchmarking exercise, including performance of the benchmarks on at two different platforms, namely V100, K80, and ICX-based systems.

|  |
| --- |
| 1. **Lessons and Summary** |

In our experience, Case 2, albeit being less complex compared to Case 1, is very ideal for automated testing. One of the crucial issues is setting up the datastore and distributing large datasets. Although we made every effort to make use of existing infrastructures, we resorted to a solution which involved setting up our own, but AWS-compatible object storage where the data is available independently and on the long-run. Although this short project was useful to gain some insights into training and inferencing, the full-scale study requires considerable amount of effort, not only to cover different platforms, and explorations, but also to fully understand and report the performance results.

⬛