Inference-as-a-Service Prototype at NERSC

Colin Thomas
University of Notre Dame
Notre Dame, Indiana, USA

Po-Han Huang
Georgia Institute of Technology
Atlanta, Georgia, USA

Hilary Utaegbulam
University of Rochester
Rochester, New York, USA

Pengfei Ding
Lawrence Berkeley National Laboratory
Berkeley, California, USA

Andrew Naylor
Lawrence Berkeley National Laboratory
Berkeley, California, USA

Xiangyang Ju
Lawrence Berkeley National Laboratory
Berkeley, California, USA

Michael Wang
Fermi National Accelerator Laboratory
Batavia, Illinois, USA

Bruno Coimbra
Fermi National Accelerator Laboratory
Batavia, Illinois, USA

Johannes Blaschke
Lawrence Berkeley National Laboratory
Berkeley, California, USA

Abstract

The increasing scale and complexity of scientific experiments has led to a growing need for efficient and scalable machine learning model inference serving systems. High-energy physics experiments and simulations of complex climate models involve petabytes of data and massive amounts of computational resources to produce accurate results. Thus, scientists are increasingly turning to utilize ML techniques to analyze and interpret the vast amount of data generated by these experiments. However, the deployment of ML models in scientific applications poses significant challenges. Traditional approaches to deploying ML models by individual users with local resources or small clusters often suffer from long startup costs and inefficient resource utilization. To address this challenge, we present a prototyped system that provides on-demand inference serving capabilities for multiple scientific ML models. Our system is deployed across the NERSC Perlmutter Supercomputer and the NERSC k8s cluster, enabling on-demand scalability.

Keywords

IaaS, HPC, Cloud Computing, HEP, Scientific Computing

ACM Reference Format:
Colin Thomas, Po-Han Huang, Hilary Utaegbulam, Pengfei Ding, Andrew Naylor, Xiangyang Ju, Michael Wang, Bruno Coimbra, and Johannes Blaschke. 2025. Inference-as-a-Service Prototype at NERSC. In . ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Summary

The increasing scale and complexity of scientific experiments has led to a growing need for efficient and scalable machine learning model inference serving systems. High-energy physics experiments, such as those conducted at the Large Hadron Collider (LHC) generate petabytes of data per year, while simulations of complex climate models require massive amounts of computational resources to produce accurate predictions. As a result, scientists are increasingly turning to utilize ML techniques to analyze and interpret the vast amount of data generated by these experiments.

However, the deployment of ML models in scientific applications poses significant challenges. Traditional approaches to deploying ML models by individual users with local resources or small clusters are often insufficient to handle the requirements of modern scientific applications, and suffer from long startup costs and inefficient resource utilization. To address this challenge, we present a prototyped system that provides on-demand inference serving capabilities for multiple scientific ML models, accessible to many concurrent users.

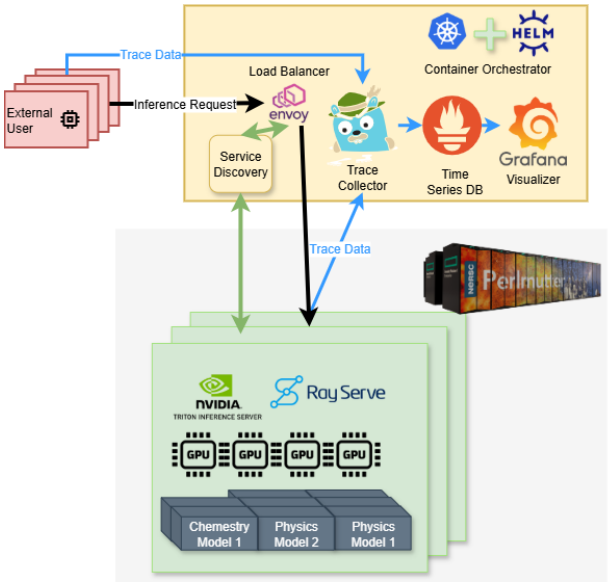

Figure 1: System architecture illustrating interactions between clients, edge services and HPC resources.

Conference'17, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YYYY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

Our system is deployed across the NERSC Perlmutter Supercomputer and the NERSC k8s cluster, as shown in Figure 1, enabling on-demand scalability. We demonstrate the functionality of our system using two Graph Neural-Network based particle tracking applications for ATLAS and DUNE, showcasing its ability to serve both applications simultaneously. To optimize inference throughput for scientific models, we conducted a comprehensive benchmarking study of NVIDIA Triton and Ray Serve under various configurations and client scenarios. In our tests, Ray Serve outperformed Triton on Python-based modules like nuGraph, whereas Triton achieved higher throughput and scales more linearly when benchmarked with perf_analyzer and a PyTorch model such as ResNet-50.

Building on the principles of Inference-as-a-Service (IaaS), we utilized the Perlmutter supercomputer to provide a lightweight, high-performance inference service tailored for scientific workloads. Our evaluation indicates that Triton achieves higher throughput and GPU utilization under moderate optimizations, such as leveraging appropriate batch sizes and efficient data serialization via the Triton Client gRPC SDK. In contrast, for workloads comprising heterogeneous tasks or relying on raw HTTP clients, Ray Serve provides more flexible scheduling and frequently attains higher throughput and GPU utilization.

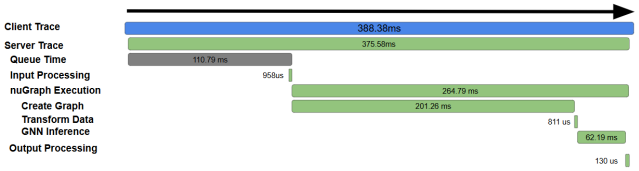

Figure 2: Time-series visualization from Jaeger

The system is instrumented with the OpenTelemetry SDK to enable detailed tracing and metrics collection on both the client and server sides. A consistent context label is applied across components, allowing precise measurement of each execution step. In the event of a performance degradation, Figure 2 shows labeled trace offers a fine-grained breakdown of the workflow. It can outline the timespan of client request initiation, preprocessing, and model inference, to bottleneck identification. Time-series metrics and trace data are collected using Prometheus and Jaeger, respectively, and are visualized through integrated Grafana and Jaeger dashboards. This setup allows efficient identification of bottlenecks and detection of idle resource utilization.

The inferences are routed to the correct model through Envoy proxy, a configurable HTTP proxy distributing the load across multiple servers as needed and providing further information about the current load, useful for the purposes of scaling up and down the allocated compute resources. Scaling policy is determined through load data through Envoy describing the quantity of inferences being made to particular models, as well as metrics from each inference server node describing saturation of resources indicating whether the addition of more compute nodes is appropriate. Clients and inference server nodes each register with a dedicated service discovery process which maintains a list of active nodes in the system for metrics scraping and a list provided to Envoy proxy identifying the active inference servers which are ready to receive inferences.

The described system offers a scalable and efficient solution for serving scientific ML models to distributed clients. Scientists benefit from reduced startup cost and shared resources, while Our work provides valuable insights into the performance of different inference serving frameworks, optimization strategies, and contributes to the development of high-performance inference services for scientific workloads, enabling researchers to focus on data analysis and discovery.