FERMILAB-PUB-22-944-ND-PPD-SCD
Comput.Softw.Big Sci.
(accepted)

Accelerating Machine Learning Inference with GPUs in ProtoDUNE Data Processing

Tejin Cai[1], Kenneth Herner[2*], Tingjun Yang[2], Michael Wang[2], Maria Acosta Flechas[2], Philip Harris[3], Burt Holzman[2], Kevin Pedro[2] and Nhan Tran[2]

[1]Department of Physics and Astronomy, York University, 4700 Keele Street, Toronto, M3J 1P3, ON, Canada.
[2]Fermi National Accelerator Laboratory, Kirk Road and Pine Streets, Batavia, 60510, IL, USA.
[3]Department of Physics, Massachusetts Institute of Technology, 77 Massachusetts Avenue, Cambridge, 02139, MA, USA.

*Corresponding author(s). E-mail(s): kherner@fnal.gov;

Abstract

We study the performance of a cloud-based GPU-accelerated inference server to speed up event reconstruction in neutrino data batch jobs. Using detector data from the ProtoDUNE experiment and employing the standard DUNE grid job submission tools, we attempt to reprocess the data by running several thousand concurrent grid jobs, a rate we expect to be typical of current and future neutrino physics experiments. We process most of the dataset with the GPU version of our processing algorithm and the remainder with the CPU version for timing comparisons. We find that a 100-GPU cloud-based server is able to easily meet the processing demand, and that using the GPU version of the event processing algorithm is two times faster than processing these data with the CPU version when comparing to the newest CPUs in our sample. The amount of data transferred to the inference server during the GPU runs can overwhelm even the highest-bandwidth network switches, however, unless care is taken to observe network facility limits or otherwise distribute the jobs to multiple sites. We discuss the lessons learned from this processing campaign and several avenues for future improvements.

Keywords: machine learning, heterogeneous (CPU+GPU) computing, GPU (graphics processing unit), particle physics, cloud computing (SaaS), neutrino physics, distributed computing

1 Introduction

Machine learning (ML)-based algorithms have been widely used in the field of neutrino physics, for applications ranging from data acquisition to data reconstruction and analysis [1–4]. A detector technology ideally suited for computer vision applications in neutrino physics is that of liquid argon time projection chambers (LArTPCs), which are employed by the Deep Underground Neutrino Experiment (DUNE) [5] and Short-Baseline Neutrino [6] experiments. ML applications are now deeply integrated into the event reconstruction and data analyses for the LArTPC experiments [7–9].

The basic unit of data is a trigger record, also known as an event or event record, which consists of a series of time samples of detector readout channels at a fixed interval within a total specified time window. The number of channels, sampling rate, and readout window vary by experiment. Event record sizes for the current generation of LArTPC experiments are typically $\leq$1 GB and are expected to increase in the next few years. With increased event size, the event reconstruction, especially the inference of ML algorithms, will become a challenge. Additionally, neutrino detectors are sensitive to neutrinos from a core-collapse supernova in or near the Milky Way. One of DUNE's physics goals is to rapidly reconstruct detector trigger records from such a supernova to provide rapid localization information to optical telescopes, placing a premium on short event reconstruction times. We have demonstrated GPU-accelerated ML inference as a service, which significantly reduced the reconstruction time for simulated neutrino events in the ProtoDUNE experiment [10]. Later, we tested the same GPU-as-a-Service (GPUaaS) approach to process the entire ProtoDUNE Run I dataset to demonstrate the scalability of this method. This paper reports the results of those tests.

2 Infrastructure setup and methods

2.1 ProtoDUNE description

The ProtoDUNE single phase detector (ProtoDUNE-SP) [11, 12] is a liquid argon time projection chamber (LArTPC) that serves as a prototype for the first far detector module of DUNE [5]. The ProtoDUNE-SP is installed at the CERN Neutrino Platform [13]. It has an active volume of $7.2 \times 6.1 \times 7.0$ m^3. The TPC wires are read out by 15,360 electric channels at a rate of 2 MHz. A typical event record consists of 6000 time samples, corresponding to a 3 ms time window. Between October 10 and November 11, 2018, ProtoDUNE-SP was exposed to a beam that delivers charged pions, kaons, protons, muons and electrons with momenta in the range 0.3 GeV/c to 7 GeV/c. After the beam runs ended, ProtoDUNE-SP continued to collect cosmic ray and calibration data until July 20, 2020, after which the detector decommissioning started. The total number of trigger records during the beam period, which consist of both beam interactions and non-beam interactions such as cosmic rays, is approximately 7.2 million.

A ProtoDUNE-SP TPC waveform recorded by a single electric channel consists of both signals and noise. There are typically three sources of signals. During the beam runs, the beam particles can interact with the liquid argon inside the TPC and produce both ionization electrons and scintillation light. Since ProtoDUNE-SP is located on the Earth's surface, it is subject to a large flux of cosmic ray muons, which induce signals over the entire detector. There are also radioactive backgrounds such as ^{39}Ar that generate low energy signals on the scale of a few hundred keV to a few MeV. Figure 1 shows the event display of a 6 GeV/c pion interaction in the ProtoDUNE-SP detector.

The first step in the reconstruction of events in the TPC is the signal processing. The goal of this stage is to produce distributions of charge arrival times and positions given the input TPC waveforms. The effects of induced currents due to drifting and collecting charge, as well as the response of the front-end electronics, are removed through de-convolution. The charge arrival distributions are used in subsequent reconstruction steps, starting with hit finding. The hit finding algorithm fits peaks in the wire waveforms, where a hit represents a charge deposition on a single wire at a given time. Each hit corresponds to a fitted peak. The hits are input to pattern recognition algorithms

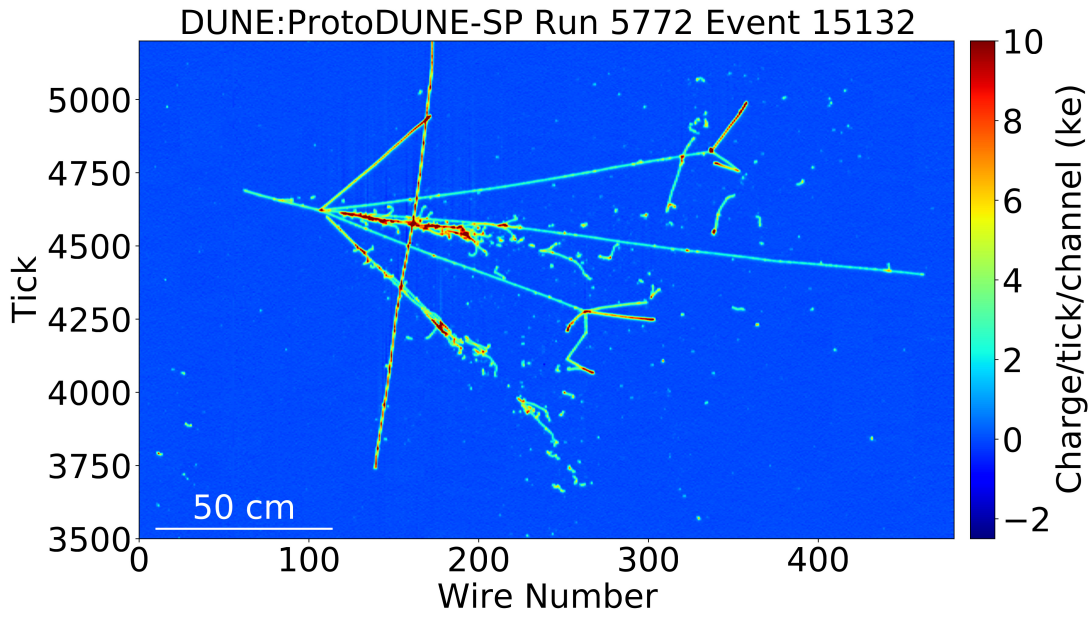

Fig. 1: A 6 GeV/c beam π^+ interaction in the ProtoDUNE-SP detector [11]. The x axis shows the wire number. The y axis shows the time tick in the unit of 0.5 μs. The color scale represents the charge deposition.

such as Pandora [14–16]. This stage finds the high-level objects associated with particles, like tracks, showers, and vertices, and assembles them into a hierarchy of parent-daughter nodes that ultimately point back to the candidate neutrino interaction. More details on the reconstruction workflow are described in Ref. [11].

In ProtoDUNE-SP, a novel algorithm is developed based on a convolutional neural network (CNN) to perform the classification of each reconstructed hit as track-like or arising from electromagnetic cascades [9]. These hit-level classifications can be used alongside pattern recognition based reconstruction algorithms such as Pandora to refine the track or shower classification of reconstructed particles. The CNN model was trained using TensorFlow [17]. In the DUNE code base this algorithm is known as *EmTrkMichelId*; hereafter, we call this algorithm EmTrk.

In order to improve the efficiency and speed of the inference of ML algorithms in a large-scale data processing, GPU acceleration specifically for the ProtoDUNE-SP reconstruction chain has been integrated without disrupting the native computing workflow using the services for optimized network inference on coprocessors (SONIC) approach [10, 18]. With the integrated framework, the most time-consuming task, track and particle shower hit identification, runs faster by a factor of 17. This results in a factor of 2.7 reduction in the total processing time when compared with CPU-only production. This initial test using a small number of simulated ProtoDUNE-SP events showed a viable, cost-effective way to solve the computing challenge facing the neutrino experiments. In this work, we report the results of reprocessing the entire 7 million ProtoDUNE-SP events taken during the test beam runs with the SONIC-enabled framework.

2.2 Inference server setup

The Nvidia Triton™ Inference Server is an open-source inference serving software that helps standardize model deployment and execution; its goal is to deliver fast and scalable AI in production [19]. NVIDIA provides multiple ways to deploy the inference server on different cloud providers and infrastructure types, including both bare metal and containerized workloads.

This study uses a cloud-based deployment of Nvidia Triton™ Inference Server within a Google Cloud Kubernetes Engine [20] cluster on virtual infrastructure provided by Google Cloud Platform. The use of this technology enables us to deploy a flexible GPUaaS model where a public endpoint takes remote inference requests from various geographically distributed sources as depicted in Figure 2. The Triton™ server running on the Google cloud supports different backends. We use the TensorFlow (version 1.15.5) backend for the inference of the EmTrk algorithm.

In a similar way as Ref. [10], this study uses several Triton™ servers split into separate Kubernetes deployments with common services for networking and external load balancing in the form of ingress objects [21]. One significant improvement for the current study is the deployment of metrics and monitoring which provided us with observability within the system in different states. In IT and cloud computing, observability is the ability to measure a system's current state based on the data it generates, such as logs, metrics, and traces. It relies on telemetry derived from instrumentation that comes from the endpoints and services in computing environments. Triton™ provides a built-in metrics endpoint [22] that publishes plain-text data in Prometheus format [23].

2.3 Methods

The DUNE collaboration undertook a production campaign in 2021 to process ProtoDUNE-SP data using the LArSoft toolkit [24] version v09_30_00. Each production run during the beam period comprises several data files, each containing between 100 and 150 data records. In contrast to the previous work, in which DUNE simulation events were processed by submitting jobs locally to a dedicated queue, we submit jobs to process each file via the current standard DUNE workflow management and job submission systems [25, 26], thus requiring no special treatment. Jobs may run either at Fermilab or one of several remote sites that we reach with opportunistic access enabled by the OSG Consortium [27].

We begin from the existing reconstructed outputs and apply the updated EmTrk algorithm to produce new outputs. Of the 7.2 million ProtoDUNE events during the 2018 beam period, we process 6.4 million through the SONIC infrastructure, and 800k with the CPU-only version of the same algorithm for comparison. The OSG sites included in the SONIC runs were chosen to be geographically proximate to the location of the Google Cloud GPU servers (which were in Iowa, USA at the time) in order to minimize the latency in data transmissions. Latency between the sites and Google Cloud server as measured by the ping utility was typically between 15 and 20 ms.

The difference in the time spent in the inference step is the primary metric with which we assess the advantage of GPUaaS over traditional CPU processing. Each job produces a log file that statistically summarizes the time spent on each stage of the event reconstruction for the job as a whole. The log has no record of per-stage processing time at the individual event level, but we can closely approximate it by taking the difference between the start times of consecutive events. We estimate

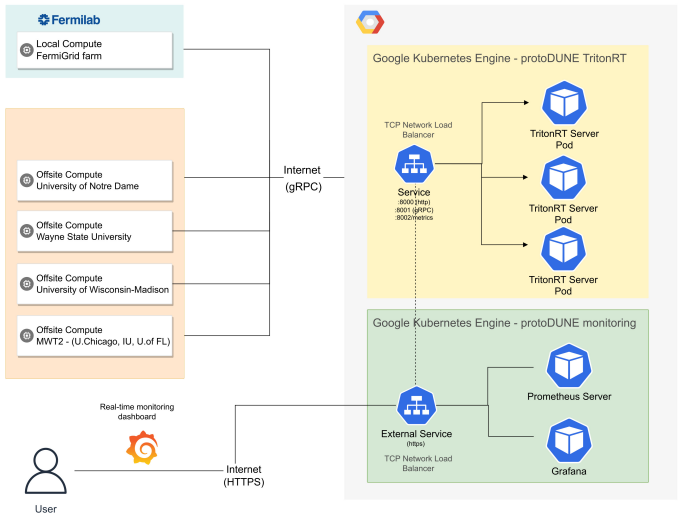

Fig. 2: ProtoDUNE GPUaaS component diagram depicting local and remote batch inference runs submitted from Fermilab and OSG Grid sites.

the per-event EmTrk duration by subtracting the median non-EmTrk duration from the total event duration, as the non-EmTrk stages display very little time variation across events. The CNN-based hit classification occurs in the EmTrk stage and is the most time-consuming step in the event reconstruction, typically accounting for more than 90% of the processing time.

3 Results

3.1 CPU-only runs

We process a set of 13 runs using CPU-based TensorFlow both at Fermilab and several off-site locations. The off-site locations are the University of Notre Dame, the University of Victoria, and the high performance computing center at Wayne State University. The TensorFlow version used in the CPU-only runs is 2.3.1. Although the TensorFlow version differs from that used in the backend for the GPU runs, the main differences between the two versions concern additional support for advanced CPU instruction sets. We therefore do not expect any significant performance differences between the two versions in the GPU case. Table 1 summarizes the number of events processed at each site and the median processing times. We did not request any specific CPU type when submitting these jobs since typical DUNE practice is to use any and all available CPU types.

Table 1: List of CPU-only run sites and median processing time

OSG Site	N samples	Median processing time (s)
FermiGrid	746603	79
Notre Dame	36082	68
Victoria	10944	52
Wayne State	4242	45

There is a clear dependence on processor type in the EmTrk processing time distribution. In general, more recent CPUs process events faster. Figure 3 shows the CPU-based EmTrk timing for each of the CPU types currently available on the Fermilab general purpose batch farm. We do not have access to CPU type information outside of Fermilab and thus group them together.

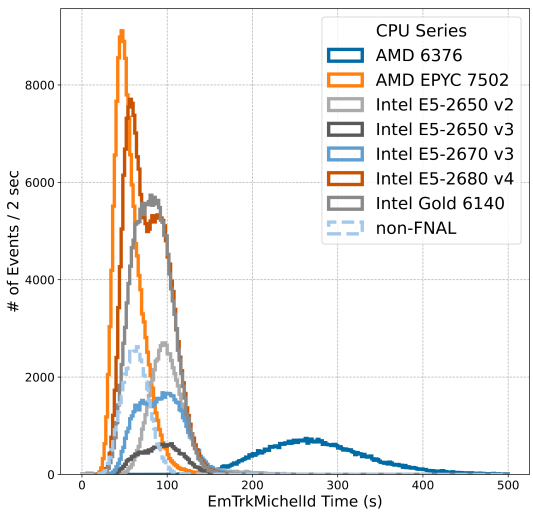

Fig. 3: Timing distributions for CPU-only runs, broken down by CPU type.

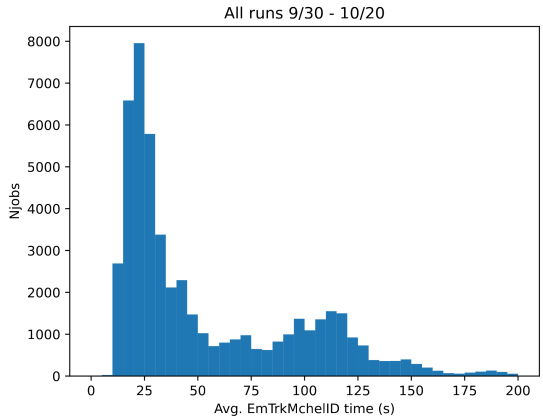

Fig. 4: Average EmTrk times for GPU runs during the period September 30, 2021 to October 20, 2021. The double peak structure arises from periods during which the outbound network connection from the Fermilab grid processing center was saturated.

3.2 GPU runs

Our main processing effort uses the GPUaaS infrastructure as described. Figure 4 shows the average EmTrk processing time when using the GPUaaS infrastructure for our entire running period. The first peak at approximately 20 s represents a factor of two improvement with respect to the fastest CPU-only runs, and a factor of roughly 11 over the slowest CPU runs. It is important to note that the EmTrk times we report here are wall times measured within the job, and thus include contributions from network latency to and from the server. There is another peak in the distribution with a median of over 100 s, to which we now turn.

3.2.1 Outbound network saturation

During the first period of GPU running we averaged between 200 and 2000 concurrent jobs. Figure 5 shows the overlay of network traffic and event processing start rate during the period of September 30, 2021 to October 6, 2021. As the event start rate increases because of the rise in the number of concurrent jobs, we see that the 100 Gb/s outbound network connection used by the Fermilab data center where the jobs run becomes saturated. While our jobs were not solely responsible for the saturation (the connection serves the entire cluster), the saturation did result in a significant increase in the average EmTrk processing time as shown in Figure 6. The highest job concurrency levels were on October 5, when unusually low demand for computing resources from other Fermilab experiments resulted in a large number of opportunistic job slots being available at Fermilab. We were, without any direct intervention, thus able to scale up to approximately 6,000 concurrent jobs. The monitoring does show switch saturation as early as October 1, however. After learning of the network saturation we implemented a concurrency limit on jobs of approximately 600; thereafter the jobs ran without incident and the EmTrk times returned to pre-saturation levels (see Figure 7).

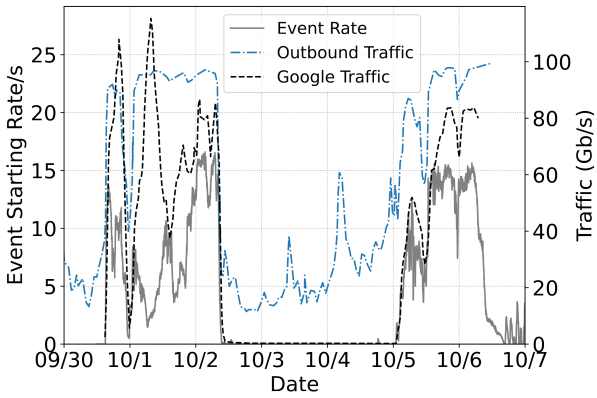

Fig. 5: Overlay of network traffic and event processing start rate at FermiGrid as a function of time, which is a proxy for the number of concurrent jobs. The origin day is September 30, 2021. The solid line is the event start rate, the blue dot-dash line is the outbound network traffic rate through the 100 Gb/s switch at Fermilab used by the batch processing cluster, and the black dashed line is the ingress rate to the Google cloud server. We are unable to disambiguate traffic sources through the switch, so the blue dot-dash line represents the total traffic as opposed to only traffic generated by our processing campaign. We see that the network switch was effectively saturated in multiple instances, though Google ingress was not.

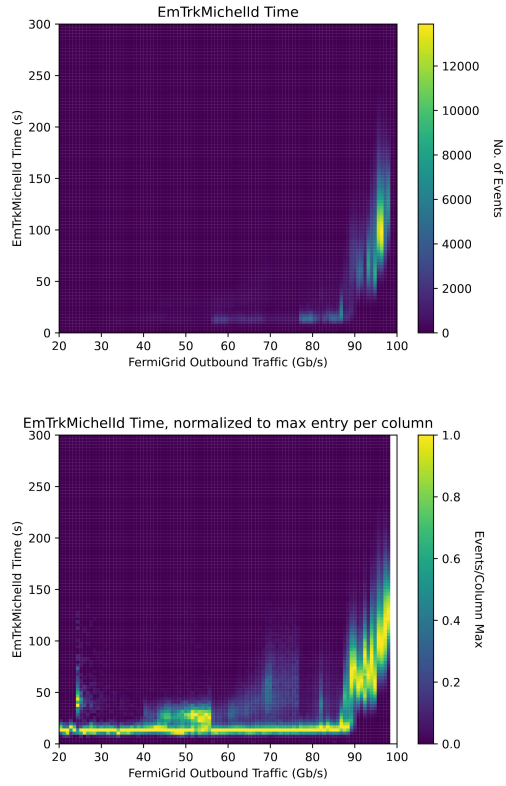

Fig. 6: The average EmTrk duration before Oct. 7 as a function of the total network traffic through the 100 Gb/s network switch at Fermilab used by the batch processing cluster. The top plot shows the real event rate. The bottom plot is the same as the top one, with each column scaled separately so the maximum amplitude is 1 for each column.

4 Discussion

In order to understand the impact of ProtoDUNE jobs on the Fermilab network traffic, we plot the distribution of event processing start rate versus network traffic in Figure 8. Even though the network traffic has contributions from all grid jobs at Fermilab, there is a clear correlation between the number of ProtoDUNE concurrent jobs and the increase of network traffic. We fit a straight line to the data points below the network traffic of 80 Gb/s. The slope of the best fit line is 4.2 ± 0.2 Gb, which is the average outbound data transmission per event. The intercept is 44 ± 2 Gb/s, which is the average traffic from non-ProtoDUNE grid jobs. Based on the discussion of transmission time in Ref. [10], for 55,000 inferences per event, with each input a 48×48 image at 32 bits, the total amount of data transmitted is about 4.1 Gigabits per event. This is consistent with the slope of the best fit straight line. The spread in data with respect to the straight line could be caused by the variation in the number of non-ProtoDUNE grid jobs during this period.

Figure 7 indicates that the average processing time is roughly 25 s/event for the GPU jobs. Assuming the entire 100 Gb/s bandwidth is available to the ProtoDUNE jobs, the maximum number of concurrent ProtoDUNE jobs we can run without saturating the network is

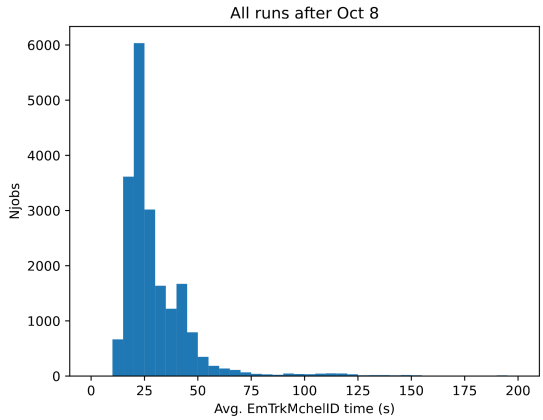

Fig. 7: The average time spent in the EmTrk task for all GPU jobs after October 8, when the network saturation had subsided.

(100 Gb/s)/(4.1 Gb/event) · (25 s/event) $\simeq$ 600. This is consistent with the concurrency limit of 600 jobs that we implemented after October 7.

Based on the above discussions, we conclude that, while overall computational time clearly decreases using GPUaaS, one does have to take particular care to understand what the expected data movement requirements will be for jobs using this architecture, and to set job concurrency limits appropriate to the capabilities of each local computing site and input data source. HTCondor [28, 29] in particular has the ability to define an arbitrary kind of resource that each job requires; one could define a "bandwidth" resource for these jobs, for example. HTCondor additionally allows configuring the job submissions to prevent more jobs to start at a given site once the sum of consumed resources by running jobs at that site reaches a certain threshold. Therefore, if one knows the total network capacity of each site hosting jobs, one can configure per-site job limits and prevent network saturation in an automated way.

4.1 Future improvements

A number of improvements to overall scalability and ease of use are possible. In addition to automatic job concurrency limits to prevent network saturation as previously described, we are exploring the possibility of compressing the data sent to the GPU server to reduce the overall bandwidth requirements. While a reduced payload would obviously increase job concurrency limits, that must be balanced against the additional run time that would be introduced in compressing and decompressing the data on the worker node and server, respectively. Another desirable area of improvement is in overall ease of use and human effort requirements. In the current setup we make use of the standard DUNE Production job submission infrastructure, which allows for a high degree of automated job submission, but due to the current nature of the cloud server it requires an authorized individual to manually instantiate the GPU inference server before we submit jobs. Establishing a method of automatically instantiating the server at job submission time and automatically ramping it down when the associated jobs are complete would avoid a clear possible failure point should no authorized individuals be available when the infrastructure is needed.

A second option to study is to use several geographically distributed inference servers instead of a single server, while also spreading the job workload over a much broader range of sites. Expanding the site pool has the advantage of making it much less likely that any single site would get enough work assigned to saturate its external connectivity, and using several inference servers spread around the world would help to mitigate the potential problem of network latency becoming comparable to the inference time. The cost changes in this scenario (for example, the relative cost of three cloud servers versus a single server three times the size)

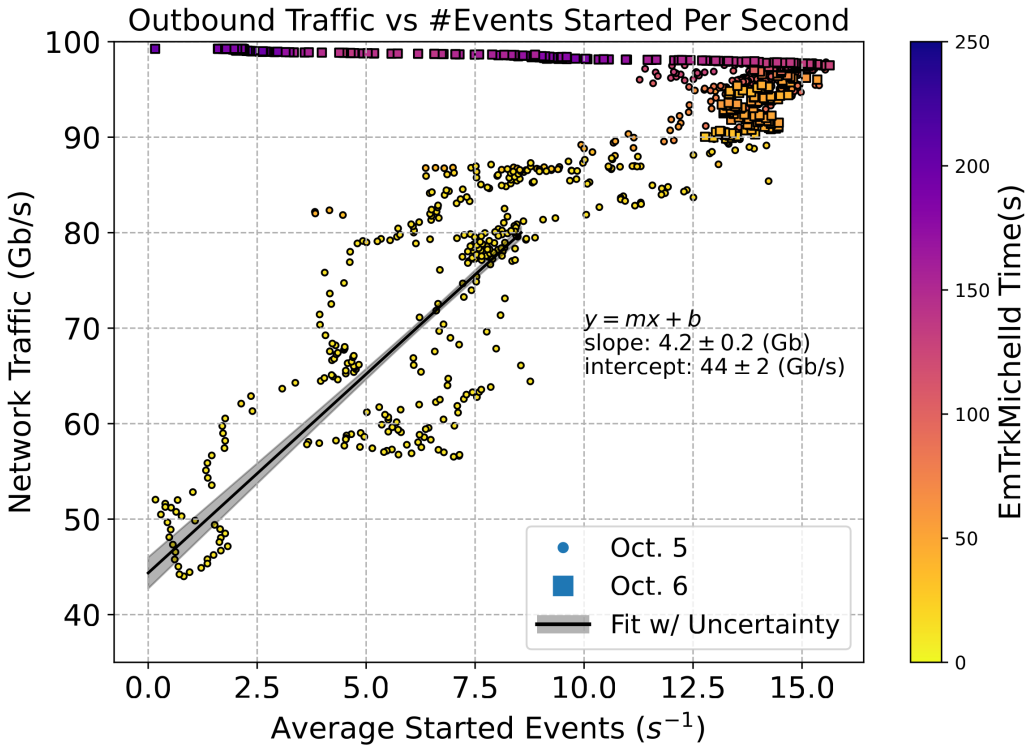

Fig. 8: The outbound network traffic vs. the average event start rate per second in 2-minute sliding windows, on October 5 and October 6. Data from each day is denoted with a different marker type. The color coding corresponds to the median EmTrk time for events in each sliding window. The linear fit to the traffic below 80 Gb/s indicates that each event sends 4.2 ± 0.2 Gb of outbound traffic, on top of 44 ± 2 Gb/s of baseline traffic from non-ProtoDUNE sources.

must be assessed and taken into account. Another consideration is how the overall event processing times would change if the worker nodes were much more geographically diffuse than they were for this study. Since we stream the input data over the network, longer network paths between the worker nodes and input data sources may lead to the non-EmTrk portions of the event processing taking longer, which in turn affects the total event processing time. DUNE is able to distribute data to various storage elements around the world via the Rucio framework [30], and pre-placing the data of interest at storage elements close to the sites to be used for processing may mitigate such concerns, though it is not required.

Another potential avenue is to use the GPU server infrastructure, but to use sites with GPUs available on the worker nodes, and run an independent server on each worker node. Several high-performance computing sites have built or are building clusters with readily available GPUs, and in some cases with multiple GPUs on each worker node, that would naturally lend themselves to such a setup. If the jobs run on worker nodes with local GPUs, external network connectivity limitations become unimportant for carrying out the inference calculations. In fact, Triton™ allows the use of shared memory for direct data transfer between CPU and GPU when the GPU is local. While it

may not be necessary to retain the server infrastructure in these cases, the advantage of doing so is that the experiment software does not have to be modified to directly access the GPU, making it maximally portable and easier to maintain. We plan to conduct a similar study using this type of setup in the future.

5 Summary

We have reprocessed approximately seven million data events from the ProtoDUNE detector installed at CERN. We use an Nvidia Triton™ inference server hosted on the Google Cloud Platform to run the most computationally expensive step of the workflow on a GPU, speeding up the required processing time by more than a factor of two, even comparing to the fastest CPU runs. Running at a scale similar to that expected during regular ProtoDUNE-II and DUNE operations, we see the expected performance improvement until the network switch through which the majority of our jobs communicate becomes saturated. Despite that, the cloud infrastructure easily kept up with demand and demonstrates the viability of the GPUaaS model at a level sufficient for current and future high-energy physics experiments, as long as the job concurrency levels at each site respect the site's network resource limits. With several promising avenues of improvement to explore, we expect that this computing model will become even more capable and easier to use in the future.

Author Contributions

All authors contributed to the study conception and design. Material preparation, data collection and analysis were performed by Tejin Cai, Kenneth Herner, and Tingjun Yang. The first draft of the manuscript was prepared by Tejin Cai, Maria Acosta Flechas, Kenneth Herner, Kevin Pedro, Nhan Tran, and Tingjun Yang. All authors read and approved the final manuscript.

Acknowledgments

We acknowledge the Fast Machine Learning collective as an open community of multi-domain experts and collaborators. This community was important for the development of this project. We acknowledge the DUNE collaboration for providing the ProtoDUNE-SP code base and data samples. The analysis is enabled in part by the Digital Research Alliance of Canada.

Declarations

Competing Interests

The authors have no competing interests to declare that are relevant to the content of this article.

Data Availability

The datasets generated during and/or analysed during the current study are available from the corresponding author on reasonable request.

Funding

MF, KH, BH, KP, NT, MW, and TY are supported by Fermi Research Alliance, LLC under Contract No. DE-AC02-07CH11359 with the U.S. Department of Energy, Office of Science, Office of High Energy Physics. NT is partially supported by the U.S. Department of Energy Early Career Award. KP is partially supported by the High Velocity Artificial Intelligence grant as part of the U.S. Department of Energy High Energy Physics Computational HEP program. PH is supported by NSF grants #1934700, #193146. Cloud credits for this study were provided by Internet2 managed Exploring Cloud to accelerate Science (NSF

grant PHY-190444). TC is supported by NSERC Canada.

References

[1] Psihas, F.: The Convolutional Visual Network for Identification and Reconstruction of NOvA Events. J. Phys. Conf. Ser. **898**(7), 072053 (2017). https://doi.org/10.1088/1742-6596/898/7/072053

[2] Perdue, G.N., *et al.*: Reducing model bias in a deep learning classifier using domain adversarial neural networks in the MINERvA experiment. JINST **13**(11), 11020 (2018) https://arxiv.org/abs/1808.08332 [physics.data-an]. https://doi.org/10.1088/1748-0221/13/11/P11020

[3] Racah, E., Ko, S., Sadowski, P., Bhimji, W., Tull, C., Oh, S.-Y., Baldi, P., Prabhat: Revealing Fundamental Physics from the Daya Bay Neutrino Experiment using Deep Neural Networks (2016) https://arxiv.org/abs/1601.07621 [stat.ML]

[4] Abbasi, R., *et al.*: A Convolutional Neural Network based Cascade Reconstruction for the IceCube Neutrino Observatory. JINST **16**, 07041 (2021) https://arxiv.org/abs/2101.11589 [hep-ex]. https://doi.org/10.1088/1748-0221/16/07/P07041

[5] Abi, B., *et al.*: Deep Underground Neutrino Experiment (DUNE), Far Detector Technical Design Report, Volume I Introduction to DUNE. JINST **15**(08), 08008 (2020) https://arxiv.org/abs/2002.02967 [physics.ins-det]. https://doi.org/10.1088/1748-0221/15/08/T08008

[6] Antonello, M., et al.: A Proposal for a Three Detector Short-Baseline Neutrino Oscillation Program in the Fermilab Booster Neutrino Beam (2015) https://arxiv.org/abs/1503.01520 [physics.ins-det]

[7] Abratenko, P., *et al.*: Search for an anomalous excess of charged-current quasielastic νe interactions with the MicroBooNE experiment using Deep-Learning-based reconstruction. Phys. Rev. D **105**(11), 112003 (2022) https://arxiv.org/abs/2110.14080 [hep-ex]. https://doi.org/10.1103/PhysRevD.105.112003

[8] Acciarri, R., *et al.*: A deep-learning based raw waveform region-of-interest finder for the liquid argon time projection chamber. JINST **17**(01), 01018 (2022) https://arxiv.org/abs/2103.06391 [physics.ins-det]. https://doi.org/10.1088/1748-0221/17/01/P01018

[9] Abed Abud, A., *et al.*: Separation of track- and shower-like energy deposits in ProtoDUNE-SP using a convolutional neural network. Eur. Phys. J. C **82**(10), 903 (2022) https://arxiv.org/abs/2203.17053 [physics.ins-det]. https://doi.org/10.1140/epjc/s10052-022-10791-2

[10] Wang, M., Yang, T., Acosta Flechas, M., Harris, P., Hawks, B., Holzman, B., Knoepfel, K., Krupa, J., Pedro, K., Tran, N.: GPU-Accelerated Machine Learning Inference as a Service for Computing in Neutrino Experiments. Front. Big Data **3**, 604083 (2021) https://arxiv.org/abs/2009.04509 [physics.comp-ph]. https://doi.org/10.3389/fdata.2020.604083

[11] Abi, B., *et al.*: First results on ProtoDUNE-SP liquid argon time projection chamber performance from a beam test at the CERN Neutrino Platform. JINST **15**(12), 12004 (2020)

https://arxiv.org/abs/2007.06722 [physics.ins-det]. https://doi.org/10.1088/1748-0221/15/12/P12004

[12] Abud, A.A., *et al.*: Design, construction and operation of the ProtoDUNE-SP Liquid Argon TPC. JINST **17**(01), 01005 (2022) https://arxiv.org/abs/2108.01902 [physics.ins-det]. https://doi.org/10.1088/1748-0221/17/01/P01005

[13] Pietropaolo, F.: Review of Liquid-Argon Detectors Development at the CERN Neutrino Platform. J. Phys. Conf. Ser. **888**(1), 012038 (2017). https://doi.org/10.1088/1742-6596/888/1/012038

[14] Marshall, J.S., Thomson, M.A.: The Pandora Software Development Kit for Pattern Recognition. Eur. Phys. J. **C75**(9), 439 (2015) https://arxiv.org/abs/1506.05348 [physics.data-an]. https://doi.org/10.1140/epjc/s10052-015-3659-3

[15] Acciarri, R., *et al.*: The Pandora multi-algorithm approach to automated pattern recognition of cosmic-ray muon and neutrino events in the MicroBooNE detector. Eur. Phys. J. **C78**(1), 82 (2018) https://arxiv.org/abs/1708.03135 [hep-ex]. https://doi.org/10.1140/epjc/s10052-017-5481-6

[16] Abed Abud, A., et al.: Reconstruction of interactions in the ProtoDUNE-SP detector with Pandora (2022) https://arxiv.org/abs/2206.14521 [hep-ex]

[17] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X.: TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. Software available from tensorflow.org (2015). https://www.tensorflow.org/

[18] The LArSoft Collaboration: The larrecodnn module. https://github.com/LArSoft/larrecodnn. Accessed: 2022-10-17 (2022)

[19] NVIDIA: NVIDIA Triton Inference Server. https://developer.nvidia.com/nvidia-triton-inference-server. Accessed: 2022-05-05 (2022)

[20] Google: Google Kubernetes Engine. https://cloud.google.com/kubernetes-engine. Accessed: 2022-07-19 (2022)

[21] Google: GKE Ingress for HTTP(S) Load Balancing. https://cloud.google.com/kubernetes-engine/docs/concepts/ingress. Accessed: 2022-07-19 (2022)

[22] NVIDIA: NVIDIA Triton Inference Server - Metrics summary. https://github.com/triton-inference-server/server/blob/main/docs/user_guide/metrics.md. Accessed: 2023-03-30 (2023)

[23] Prometheus Authors: Exposition Formats. https://prometheus.io/docs/instrumenting/exposition_formats/. Accessed: 2023-03-30 (2023)

[24] Snider, E.L., Petrillo, G.: LArSoft: Toolkit for Simulation, Reconstruction and Analysis of Liquid Argon TPC Neutrino Detectors. J. Phys. Conf. Ser. **898**(4), 042057 (2017). https:

//doi.org/10.1088/1742-6596/898/4/042057

[25] Herner, K.: DUNE Production processing and workflow management software evaluation. Euro. Phys. J. Web of Conf. **245**, 03019 (2020). https://doi.org/10.1051/epjconf/202024503019

[26] Mengel, M., White, S., Podstavkov, V., Wiersma, M., Mazzacane, A., Herner, K.: Production Operations Management System (POMS) for Fermilab Experiments. European Physical Journal Web of Conferences, vol. 245, p. 03024 (2020). https://doi.org/10.1051/epjconf/202024503024

[27] Pordes, R., OSG Consortium, Petravick, D., Kramer, B., Olson, D., Livny, M., Roy, A., Avery, P., Blackburn, K., Wenaus, T., Würthwein, F., Foster, I., Gardner, R., Wilde, M., Blatecky, A., McGee, J., Quick, R.: The Open Science Grid. J. Phys. Conf, Ser. **78**, 012057 (2007). https://doi.org/10.1088/1742-6596/78/1/012057

[28] Thain, D., Tannenbaum, T., Livny, M.: Condor and the grid. In: Berman, F., Fox, G., Hey, T. (eds.) Grid Computing: Making the Global Infrastructure a Reality. John Wiley & Sons Inc., Hoboken, NJ (2002)

[29] Bockelman, B., Livny, M., Lin, B., Prelz, F.: Principles, technologies, and time: The translational journey of the HTCondor-CE. Journal of Computational Science (2020). https://doi.org/10.1016/j.jocs.2020.101213

[30] Barisits, M., Beermann, T., Berghaus, F., Bockelman, B., Bogado, J., Cameron, D., Christidis, D., Ciangottini, D., Dimitrov, G., Elsing, M., Garonne, V., di Girolamo, A., Goossens, L., Guan, W., Guenther, J., Javurek, T., Kuhn, D., Lassnig, M., Lopez, F., Magini, N., Molfetas, A., Nairz, A., Ould-Saada, F., Prenner, S., Serfon, C., Stewart, G., Vaandering, E., Vasileva, P., Vigne, R., Wegner, T.: Rucio: Scientific data management. Computing and Software for Big Science **3**(1), 11 (2019). https://doi.org/10.1007/s41781-019-0026-3