nature astronomy

PERSPECTIVE

https://doi.org/10.1038/s41550-022-01651-w

Hardware-accelerated inference for real-time gravitational-wave astronomy

Alec Gunny[1,2,3] ✉, Dylan Rankin[1,3] ✉, Jeffrey Krupa[1,3], Muhammed Saleem[4], Tri Nguyen[1,2,3], Michael Coughlin[4], Philip Harris[1,3], Erik Katsavounidis[1,2], Steven Timm[5] and Burt Holzman[5]

Computational demands in gravitational-wave astronomy are expected to at least double over the next five years. As kilometre-scale interferometers are brought to design sensitivity, real-time delivery of gravitational-wave alerts will become increasingly important to enable multimessenger follow-up. Here we discuss a novel implementation and deployment of deep learning inference for real-time data denoising and astrophysical source identification. This objective is accomplished using a generic inference-as-a-service model capable of adapting to the future needs of gravitational-wave data analysis. The implementation allows seamless incorporation of hardware accelerators and also enables the use of commercial or private as-a-service computing. Low-latency and offline computing in gravitational-wave astronomy addresses key challenges in scalability and reliability and provides a data analysis platform particularly optimized for deep learning applications.

We have entered a new era where discoveries in astronomy will be driven by combining observations in gravitational waves and the electromagnetic spectrum, as well as neutrinos[1,2]. This is often referred to as multimessenger astronomy, and it has been enabled by the direct detection of gravitational-wave transients[3,4] with the large ground-based laser interferometers LIGO (Laser Interferometer Gravitational-Wave Observatory)[5,6] and Virgo[7]. The data analyses for gravitational-wave transients of both known and unknown signal morphology present a major computational challenge for these instruments as they prepare to enter their fourth observing run (referred to as 'O4') in the summer of 2022[8]. Existing gravitational-wave detection algorithms for compact binary systems rely heavily on parameterized waveforms (templates) and the use of matched-filtering techniques[9,10]. While such approaches have performed well since the start of LIGO running, they scale poorly with the expected low-frequency improvement of instruments as well as with the expanding parameter space needed to cover spin effects and sub-solar-mass compact binaries[11]. The anticipated addition of KAGRA (Kamioka Gravitational Wave Detector)[12,13] to the international network of detectors observing the gravitational-wave sky is expected to further increase the computational demands.

At the same time, gravitational-wave transients of ill defined morphology (for example, supernovae, neutron star glitches and potentially yet unknown astrophysical systems) are susceptible to instrumental and environmental noise that is hard to simulate and often challenging to subtract[14,15]. Such noise sources will affect searches for binary systems with sufficient signal duration to make it statistically probable to overlap with non-Gaussian artefacts, as in the case of GW170817[4]. While the vast majority of data analysis techniques employed in gravitational-wave searches are built on traditional time–frequency decomposition using Fourier and wavelet transforms, machine learning (ML) techniques have recently emerged as potentially powerful solutions to the computational challenges in this field[16]. Neural networks and other gradient-based learning algorithms have been proposed in gravitational-wave analyses for tasks such as noise regression[17,18], astrophysical searches[19,20], parameter estimation[21,22] and transient noise classification[14,23]. While there remains debate on whether ML algorithms can fully match the robustness of more traditional approaches in certain domains, their current interest for the physics community and long-term potential make it important to understand how best to leverage them in practice. This debate is particularly relevant in gravitational-wave analysis, where the comparatively large computational demands make their application difficult.

Innovative hardware-based acceleration with graphics processing units (GPUs) and field programmable gate arrays (FPGAs) have recently been gaining ground within industry and academic research[24,25] as methods for performing fast ML inference at large scales. For example, Microsoft uses a large FPGA-based system to assist its Bing search engine[26], the IceCube experiment has utilized a large amount of cloud resources to accelerate its event reconstruction on GPUs[27] and computations for lattice quantum chromodynamics have made heavy use of GPUs, particularly at high-performance computing centres. However, to take full advantage of accelerators, modifications must be made to standard models of computing. An alternative model, which has gained popularity in other fields such as neutrino physics[28] (at ProtoDUNE) and collider physics[29,30] (at the Large Hadron Collider), is called 'as a service', or 'inference as a service' (IaaS) when used to specifically denote accelerated ML inference. In this IaaS model, trained ML models are hosted in a centralized repository, from which an inference application loads and exposes them to requests from networked clients. The user then requests inferences by sending packets of inputs to the server, but the details of the server are abstracted from the user using standard client application programming interfaces. This model allows the simple integration and management of heterogeneous computing resources as well as parallel and asynchronous execution of inference requests.

Inference-as-a-service framework

In the following, we will analyse the advantages of the IaaS paradigm using two deep learning models as examples of realistic ML usage in gravitational-wave analysis. The first is DeepClean[18],

[1]Department of Physics, MIT, Cambridge, MA, USA. [2]LIGO Laboratory, MIT, Cambridge, MA, USA. [3]The NSF Institute for Artificial Intelligence and Fundamental Interactions, Cambridge, MA, USA. [4]School of Physics and Astronomy, University of Minnesota, Minneapolis, MN, USA. [5]Fermi National Accelerator Laboratory, Batavia, IL, USA. ✉e-mail: alec.gunny@gmail.com; drankin@mit.edu

a one-dimensional convolutional autoencoder able to predict and subtract noise present in the gravitational-wave sensing channel (referred to as 'strain'). Gravitational-wave interferometers are subject to technical and environmental noise that ultimately limit the instruments' ability to reach their design sensitivity. The bulk of data written onto disk by these instruments corresponds to auxiliary channels that monitor and record the interferometry as well as their physical environment. These auxiliary channels allow for effective monitoring and regression of transient or continuous noise from the gravitational-wave measurement. To achieve this noise removal, about 200,000 such auxiliary channels resulting in over 10 MBps of time-series data from each interferometer are continually written onto disk during normal data acquisition[31]. We generally refer to them as 'witness' channels for their ability to record noise that may affect the measurement and their role in assisting noise subtraction from the strain channel. DeepClean uses information from the auxiliary channels correlated with the strain channel to achieve noise reduction. It can also be customized to specific noise couplings for a variety of applications[18]. In the use-case described here, we use 21 auxiliary channels that are good witnesses of noise appearing in the strain channel and associated with monitor lines of power mains and their harmonics, including sidebands.

The second model we will use to demonstrate the IaaS framework, called BBHnet (T.N. et al., manuscript in preparation), is used in the archetypal search for compact binary black hole coalescences. BBHnet utilizes one-dimensional convolutional neural networks to distinguish between binary black holes and detector noise, with the noise-regressed strain signal derived by DeepClean as its input. The combination of DeepClean and BBHnet offers an end-to-end test of a pipeline that combines both data cleaning and astrophysical searches. The task of binary black hole identification is a challenging one, but deep learning has been shown to be an effective replacement[19] for the template-matching algorithms that are currently used. The number of templates required for such algorithms can grow to the millions as searches expand to cover neutron stars and systems with sub-solar component masses[11]. Moreover, the continuing improvement of the low-frequency sensitivity of interferometers and the addition of new detectors to the international gravitational-wave network[8] also grow the template banks used in gravitational-wave searches. Deep learning is only expected to become more critical in this regime due to the large number of free parameters in the compact binary star system numerical problem. At the same time, the ability to perform such gravitational-wave searches in real time enables their follow-up via the electromagnetic spectrum and neutrinos, thus enabling multimessenger astronomy.

To deploy these pipelines, we use an out-of-the-box inference service developed by NVIDIA called the Triton Inference Server[32]. Triton supports concurrent inference execution on both CPUs and GPUs using multiple framework backends. It is provided as a containerized application to make it portable to different deployment locations. Triton automatically detects changes in the model repository from which it reads, and will update the models that it hosts in-memory in response to updates according to a prescribed versioning policy. This design ensures that all services, and their users, are kept in sync with the latest developments and with one another. This feature is particularly beneficial for computing scenarios where a distributed user base is interested in accessing centrally managed server resources, such as HTCondor and similar distributed computing tools.

Inference scenarios in gravitational-wave data analyses can be broadly separated into online and offline categories. The online inference scenario requires low-latency inferences for real-time processing of live streams during data collection runs[33,34]. The offline scenario, on the other hand, involves large-scale processing of archival data for use-cases such as model validation analyses or completion of transient event searches[35,36] and corresponding catalogues[37]

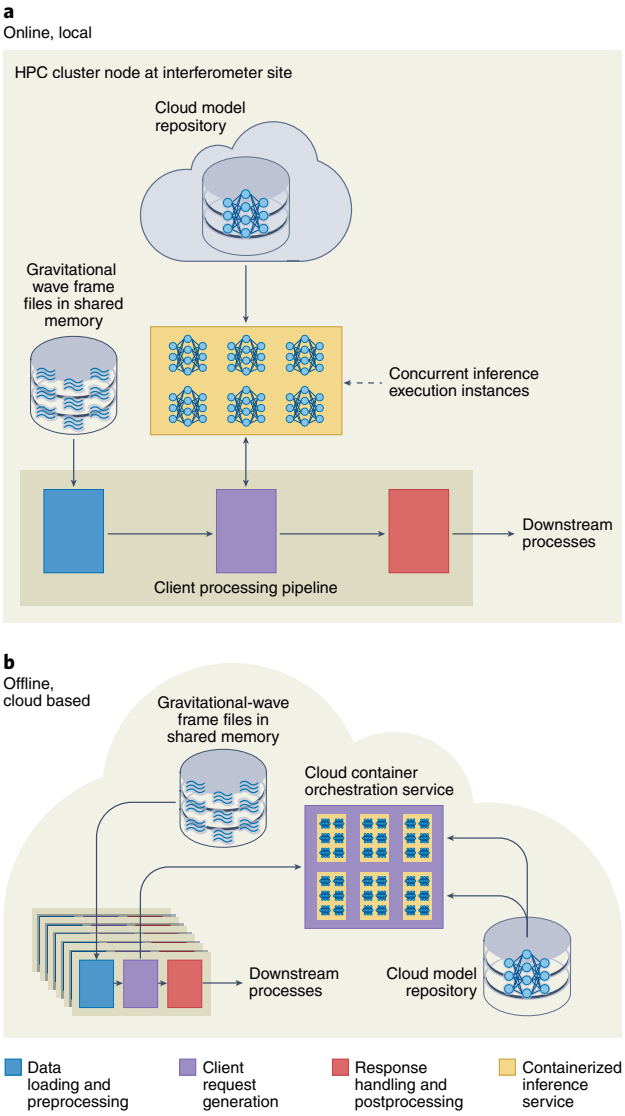

Fig. 1 | Example IaaS deployment scenarios. a, Locally (that is, at the gravitational-wave detector sites) deployed client and server instances collocated with in-memory data sources to minimize latency. The server is deployed in a container using Singularity[40] and reads from a cloud-based model repository to stay in sync with updates. **b**, Offline collocated cloud-based deployment where multiple server instances are managed by Kubernetes and data are split among multiple client virtual machines.

following the definitive calibration of the instruments. Figure 1 depicts the two IaaS deployment scenarios we have adopted for realistic online and offline use-cases in gravitational-wave experiments. Additionally, Fig. 1a depicts the deployment for an online pipeline that leverages DeepClean to remove noise from the strain channel at each detector to be made available to downstream transient-event detection algorithms in real time. To meet the low-latency requirements of multimessenger astronomy, this pipeline is deployed fully on the LIGO Data Grid (LDG: https://computing.docs.ligo.org/lscdatagridweb/) so that the data sources, client, and inference service can minimize latency incurred by networked connections. Figure 1b presents a generic offline scenario we use for two archival data processing tasks. These use-cases, unlike the online use-case, prioritize the total processing time for a given dataset instead of the individual request latency. In addition, they can be massively parallelized by breaking the dataset into smaller, time-contiguous

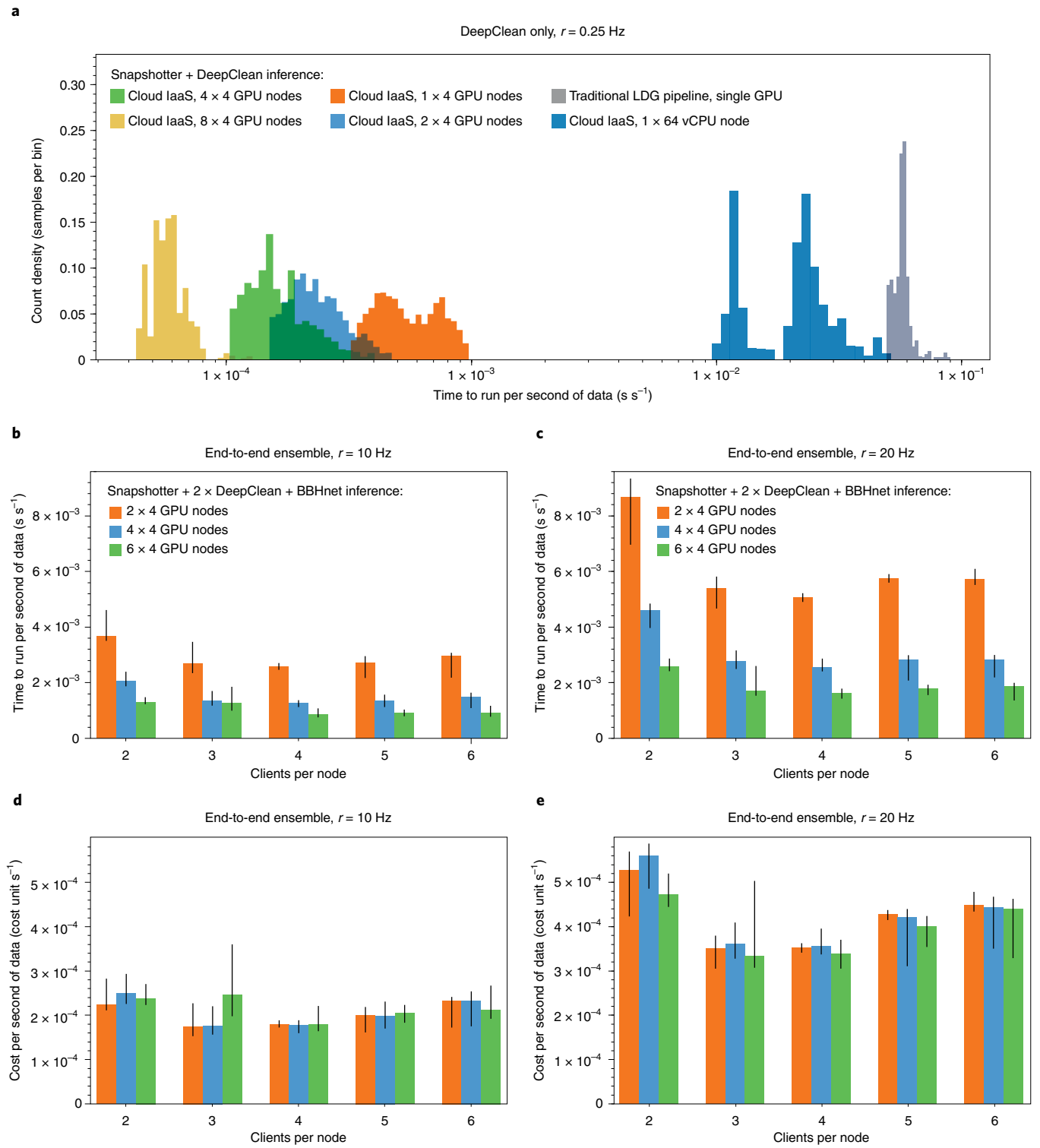

Fig. 2 | Distributions of time and cost required to process 1 s of data. a, All GPU-equipped inference service nodes used NVIDIA V100 32 GB GPUs, were equipped with 32 virtual CPUs (vCPUs) and hosted six concurrent execution instances per GPU. The CPU-only inference service hosted six concurrent execution instances for the whole node. **b**,**c**, The coloured bars represent the median time to run per second of data processed, with ±25th percentiles depicted by the black bars. **d**,**e**, The coloured bars represent the median cost per second of data processed, with ±25th percentiles depicted by the black bars. All nodes in **b**–**e** were equipped with four NVIDIA T4 GPUs and 32 vCPUs, and the number of concurrent executions per GPU was six for each DeepClean instance and one for all other models except the snapshotter.

subsets, which are assigned to individual client instances. In both these scenarios, building an optimal IaaS pipeline involves tuning the same parameters, but their different objectives and constraints lead to very different decisions for which set of values is optimal. Moreover, the streaming nature of gravitational-wave data presents unique challenges to the IaaS model in both deployment scenarios.

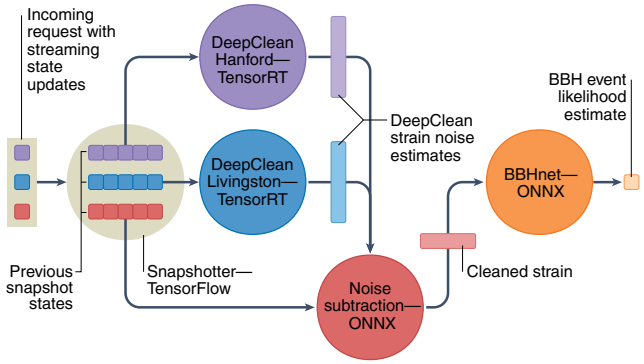

Fig. 3 | Flow of data through ensemble of individual modules on inference service in end-to-end pipeline. The snapshotter model maintains and updates state for all three data sources at once, then sends the updated snapshots to their respective downstream models. The framework backend used for each model is indicated. Using Triton's ensemble scheduler, arbitrary server-side pipelines such as this can be constructed. The ability to construct complex ensembles of otherwise disparate models implemented using many software backends illustrates the robustness of the IaaS model.

To reduce the overall bandwidth going into the inference service, previous time-series samples are cached on the GPU so that only the new samples need to be sent to the server. More details can be found in Supplementary Information.

Offline usage

As a first test of the offline use-case, we deploy DeepClean to remove noise from roughly a month's worth of strain data from the O3 observing run of the LIGO–Virgo instruments[8]. As discussed in Supplementary Information, we sample fixed-length kernels for inference from the time series at an inference sampling rate r of 0.25 Hz. Figure 2a shows the distribution of wall-clock processing time per second of data achieved both by a traditional workflow on the LDG and by workflows leveraging an inference service on various amounts of cloud resources. These workflows are described in detail in Supplementary Information. The LDG workflow, implemented on a single GPU, has the longest processing time, shown in grey. For the IaaS workflow using only CPUs we observe a reduction in the processing time by a factor of 3–5 when compared with a traditional, serially executed workflow. This reduction arises largely from the inference service's trivial parallelization of server-side inference and client-side tasks such as data loading and preprocessing, as well as from efficient scheduling of concurrent inference execution on the server. Further reductions in time are achieved by adding GPUs to the service. An inference service equipped with four GPUs is able to decrease the processing time by another factor of 10–12, and the reduction continues proportionally as more service nodes are added. This scaling to additional GPUs can be handled seamlessly in the IaaS paradigm.

The second offline workflow, an end-to-end pipeline implemented in multiple different frameworks and comprising both DeepClean and BBHnet, is shown in Fig. 3. The time and cost required to process 1 s of data using this pipeline with various server and client configurations are shown in Fig. 2b–e. The cost is computed by aggregating the cost per unit time of all client and server resources and normalizing by the cost of 1 CPU h. See Supplementary Information for more details of these measurements. As can be seen from Fig. 2b,c, for both values of r and the number of clients per server node, the processing time decreases nearly linearly as the number of server nodes is increased. However, the total cost per second of data remains nearly constant with increasing numbers of nodes leveraged by the inference service, as shown in Fig. 2d,e. These trends show that once external constraints such as cost or sampling rate are imposed the IaaS workflow is able to make efficient use of the available resources, regardless of the exact values of the constraints.

We see in Fig. 2b,c that increasing the clients per server node from two to four is able to reduce the total processing time, regardless of the actual number of server nodes. This reduction is possible because we do not fully utilize the server's resources, and therefore more clients can be served with the unused resources. This behaviour underscores the ability of off-the-shelf IaaS implementations to take full advantage of scarce server resources through trivial scaling in a way that is independent of the details of the particular algorithms being deployed. For larger numbers of clients we observe the same processing time, but an increased cost as a result of the saturated server throughput. In this particular example, the models limiting the throughput are the two DeepClean instances, which are the larger of the models in the pipeline. Some optimizations have already been applied to these models for this pipeline, but further improvement to their inference throughput is being investigated.

Online usage

The IaaS paradigm is equally capable of performing in an online setting. Unlike the offline settings described above, the online setting prioritizes low latency. DeepClean in particular has an extremely high latency sensitivity, since the frames it cleans are already a few seconds delayed due to detector calibration processes, and its cleaned outputs will be made available to downstream event detection and characterization algorithms that may themselves incur multiple seconds of latency. To be valuable in the multimessenger astronomy setting, then, we would like DeepClean to be able to clean data within milliseconds to minimize its cost to downstream applications. Figure 4 depicts the latency achieved with various server configurations as a function of r for the DeepClean pipeline. We disaggregate the latencies into time spent computing model inference (light blue) and time spent queuing for available resources (light brown). At low values of r, the latency is dominated by compute time and remains nearly constant regardless of server configuration, since a single GPU is capable of handling the request load. As r increases past ~1,000 Hz, the request load overwhelms the maximal GPU performance, and so requests must queue and wait for resources to become available. At the highest values of r, this resource availability is the primary determinant of total latency, which becomes a near-linear function of the amount of server resources. More detailed inspection of latency sources indicates that the bottleneck in this pipeline is the streaming state update described in Supplementary Information, which limits the capacity for the downstream DeepClean model to benefit from additional GPUs. Future optimizations to this update step via heterogeneous high-performance computing techniques will allow this pipeline to better utilize the available resources to both increase the values of r that can be processed stably and decrease the latency incurred at sustainable values of r.

At-scale testing

Finally, we emulate the offline end-to-end pipeline in the context of jobs running over a prolonged period of time. The jobs are submitted to compute instances on the Google Cloud Platform via the HEPCloud framework[38]. HEPCloud enables scientific experiments to access a heterogeneous variety of computing resources (on-premises batch, commercial cloud, supercomputing facilities), while presenting a single simple frontend interface (HTCondor) to the end user. HEPCloud is a good testbed for prototyping as-a-service workloads. First, it allows a large number of CPU, GPU and other resources to be provisioned at a single site,

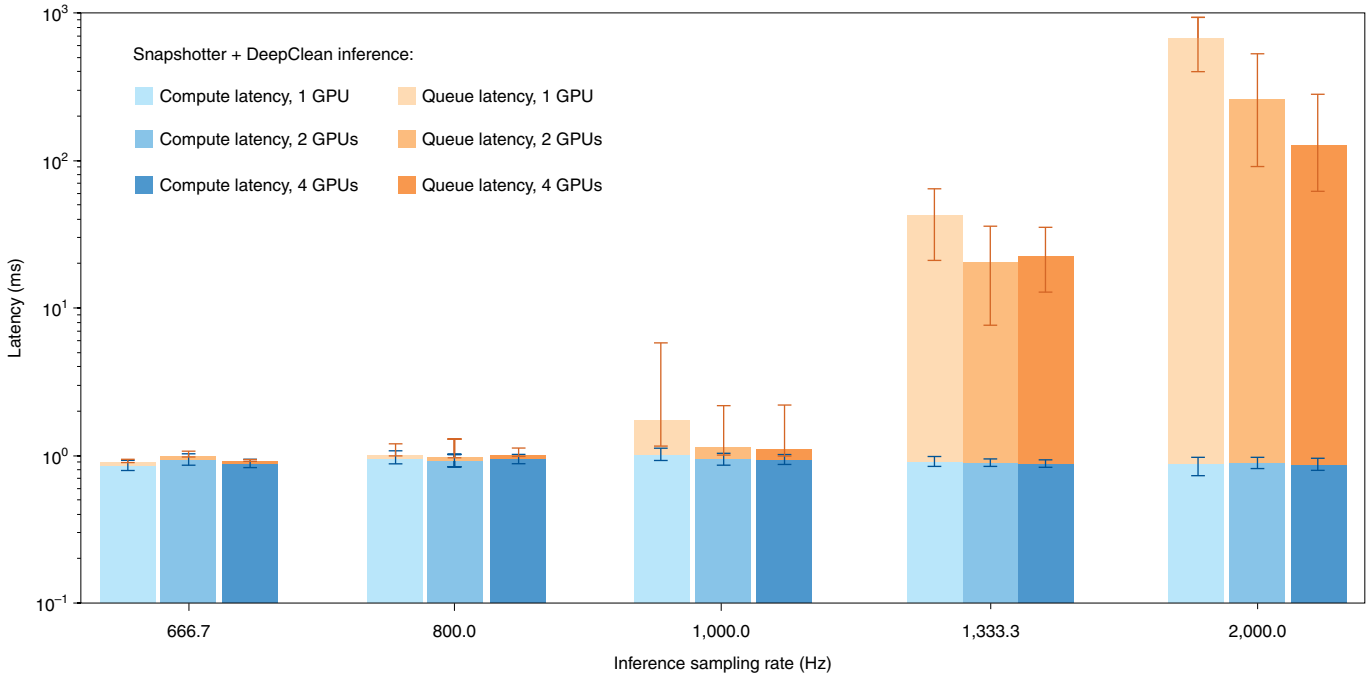

Fig. 4 | End-to-end server-side latency for multiple GPU counts as a function of *r*. Latency is broken down into contributions from compute time and queuing time, with vertical bars representing median values and error bars representing ±25th percentiles. The data sampling rate f_s was fixed at 4,000 Hz for all measurements, and all GPUs used were NVIDIA V100 16 GB GPUs.

facilitating scalable intrasite tests. Second, these resources form a self-contained system free from other jobs, which increases reproducibility for testing new frameworks. Finally, the HTCondor frontend is widely used in pre-existing job submission frameworks. The clients are configured to run on Google Cloud CPU nodes accessed via HEPCloud. The servers are configured to have between 4 and 80 NVIDIA T4 GPUs at the same site.

The sustained inference throughput as a function of time is shown in Fig. 5. With a server consisting of four GPUs, stable processing of 1,000 inferences per second (2 s of data per second) is observed. This work is distributed across 100 HEPCloud clients. The observed throughput scales linearly with the number of servers (GPUs), reaching 20,000 inferences per second (40 s of data per second) for 80 GPUs communicating with 2,000 clients. This test is a demonstration of our frameworks' ability to deliver inferences for a sustained period of time, and with large numbers of resources, using existing gravitational-wave experimental paradigms.

Current limitations

While we have shown that the IaaS paradigm is capable of meeting the computational needs of streaming low-latency data denoising and astrophysical searches, there are additional considerations that are required for a fully real-time pipeline for multimessenger follow-up. Specifically, the performance of the deep neural network inference pipeline must operate with the same fidelity in the offline scenario. In this instance, we have observed that there is some degradation in the subtraction quality at the edges of the cleaned segments when using DeepClean. In the present set-up we simply exclude the quality-degraded edges from the cleaned data segments at a latency cost equal to the excluded data length. We refer to this latency as the aggregation latency. Figure 6 demonstrates this degradation and subsequent recovery by comparing the performance with the fully offline DeepClean pipeline. We see that an aggregation latency of 0.75 s is able to closely reproduce the amplitude spectral density of offline DeepClean. While this aggregation latency limits the minimum possible latency for the online pipeline we have used, it could potentially be reduced or even removed entirely by an algorithm designed (trained) specifically for low-latency cleaning.

Another factor that must be considered when discussing the overall latency is how often the trained network must be updated on the server. Typically a cleaning algorithm must be retrained on recent data to maintain performance under gradually changing conditions. For continuous functioning, the trained model must remain valid for longer than the time it takes to retrain a new model. For DeepClean, training the network on a new dataset typically takes <20 min on a single GPU, while analysis of O3 gravitational-wave data from the LIGO–Hanford detector shows that a trained model can be used to clean at least 4,096 s of subsequent data without compromising the quality of its predictions. Since this time is well beyond the required training time already, it ensures that DeepClean can be safely employed in an environment with a non-stationary noise distribution.

We have demonstrated a fully realistic computational workflow to process gravitational-wave data with ML algorithms using a heterogeneous computing stack, using it to deploy ML algorithms for both noise subtraction and binary black hole detection. Furthermore, we have shown that we can increase the input throughput to this workflow by several orders of magnitude by caching updates to the time series under analysis. Our workflow can seamlessly incorporate future updates to either algorithm studied here, and can be extended to additional algorithms for detection or any other analysis. We have run this workflow with real gravitational-wave data and demonstrated operation in two different scenarios: online and offline data processing. We find that in the online scenario our current set-up can perform real-time noise subtraction and binary black hole detection with a latency of 1 s, despite bottlenecks introduced by a serial time-series caching step. A server with a GPU located locally at each gravitational-wave site would be able to output a cleaned stream of data within 1 s of acquisition. With modified deep neural network models, it is likely that the latency can be reduced to milliseconds. The success of this scheme is not dependent on the specifics of the algorithms

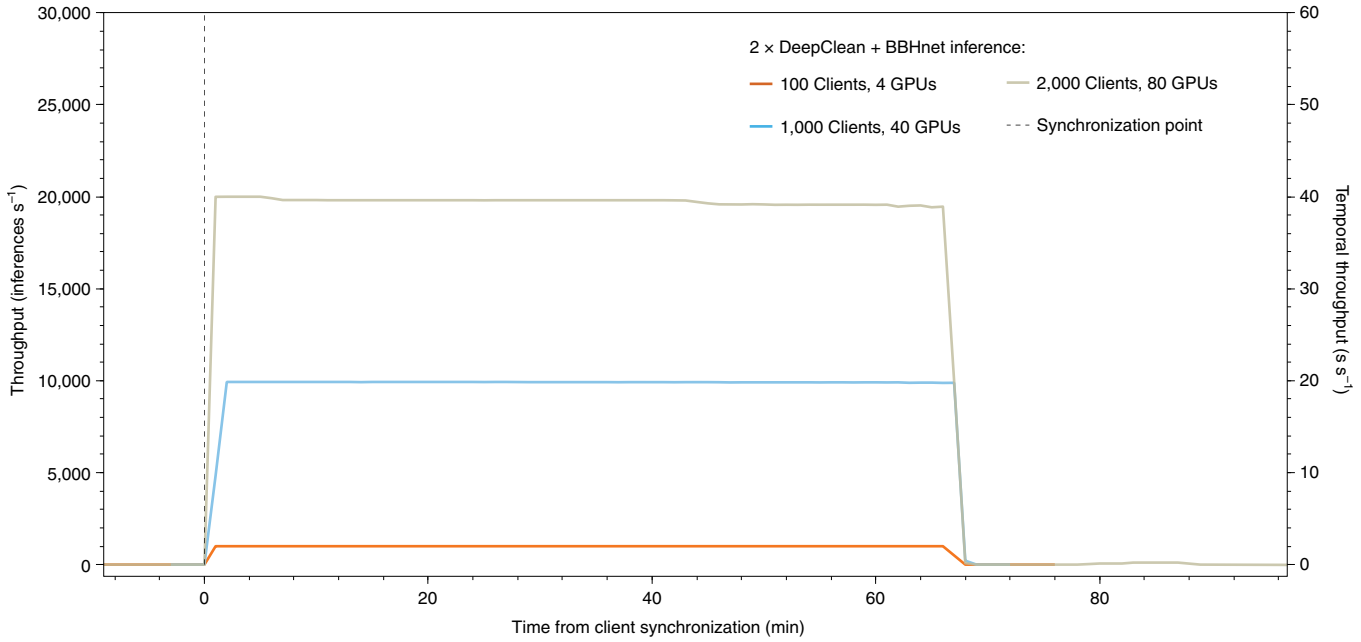

Fig. 5 | The number of seconds of gravitational-wave data processed per second as a function of time for a sustained test using HEPCloud. The jobs are synchronized to start at the time indicated by the dotted vertical line.

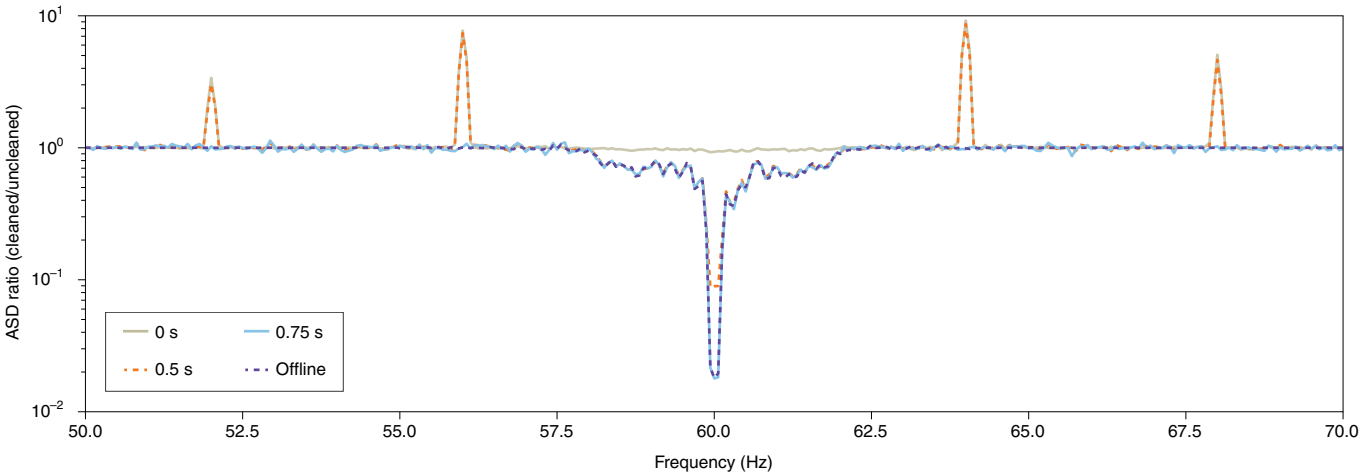

Fig. 6 | Performance of the online-deployed DeepClean noise regression as a function of the aggregation latency—length of the time-series data excluded from each cleaned segment, to avoid noisy edges. The quantity on the *y* axis is the ratio of the amplitude spectral density (ASD) of the cleaned data to that of the uncleaned data. We compare the offline case with an analysis with zero, 0.5 s and 0.75 s latency. At zero latency, the fraction of frequency bins within [50, 70] Hz that differ from the offline ASD ratio by more than 10% is ~23%; this quantity reduces to ~4% and ~1% with 0.5 s latency and 0.75 s latency respectively.

implemented, which is crucial for its long-term viability as computational needs and resources develop.

Outlook

For offline gravitational-wave data processing, we have set up a full reprocessing stream. By relying on the IaaS model, we are able to optimally configure the GPUs and CPUs to process the datastream, leading to large increases in the overall throughput of the system with respect to more standard deep learning inference implementations. For large many-GPU setups we demonstrate orders of magnitude reductions in the time required to process gravitational-wave data, and these reductions scale proportionally to the number of GPUs. This scaling is another crucial advantage of the IaaS model, particularly as computing systems evolve and are updated and expanded. Even for purely CPU-based set-ups we observe an order of magnitude reduction in the processing time with respect to a traditional, CPU-only workflow. The pipeline we employ is chosen purposefully to demonstrate this reduction for a realistic workflow. For any underlying background noise that is consistently present in the detectors and has correlation with auxiliary sensors, such as noise sources due to power lines, and its harmonics and sidebands, DeepClean is capable of cleaning the noise source. BBHNet, on the other hand, separates noise transients from gravitational-wave signals using the lack of coherence in noise transients between detectors. Taken together, these ML algorithms can identify and mitigate both omnipresent and transient noise sources affecting

gravitational-wave searches. Our system is dynamically scalable, and remains optimized whether we use a small number of GPUs or a larger number of resources. We have also demonstrated scalability by testing a sustained offline workflow with HEPCloud on Google Cloud. As a consequence, we have demonstrated that we can scale out gravitational-wave reprocessing to utilize a large number of computing nodes available within a cloud or high-performance computing centre. With sufficient computing resources, our computing scheme can reprocess gravitational-wave datasets containing many years of data within a few hours. Extrapolating from Fig. 5 we can project that processing 365 d of data in 6 h would require approximately 3,000 T4 GPUs, with this number reduced in the case of more powerful GPUs. This amount of GPUs is already within the current capabilities of cloud providers[27], although using this many GPUs often would be quite costly in the cloud. The number of CPUs needed would also be quite large (75,000 clients) to sustain the necessary request rate, but is within the current capabilities of cloud providers or large grid computing.

To conclude, this Perspective has illustrated the efficacy of an ML-based heterogeneous computing model that can integrate seamlessly within existing gravitational-wave computing stacks and is ready to be deployed. Adjustments to the ML algorithms used, or additions of new models, can be handled in a straightforward manner. Our implementation shows the ability to meet latency and scale requirements for online and offline uses in gravitational-wave ground-based interferometers such as LIGO, Virgo and KAGRA. The IaaS paradigm also has broad implications across many fields where the fast processing of real-time datastreams is critical, including areas of electromagnetic and neutrino astronomy. Most importantly, this paradigm has the potential to improve our ability to perform multimessenger astronomical observations through high-throughput and low-latency inference. It also offers a possible means with which to incorporate ML-based techniques for real-time controls of the numerous servo loops that are part of the laser interferometry in present and future ground interferometers[39]. Substantial work remains to realize these possibilities. As gravitational-wave detectors become increasingly sensitive over the course of second-generation improvements in this decade[8], and with third-generation improvements in the next[39], this new heterogeneous computational stack has the ability to facilitate the computational demands needed to accelerate discovery.

Data availability

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

Code availability

All code used to generate the data for both the online and offline experiments in this study is available at https://github.com/fastmachinelearning/gw-iaas.

Received: 21 September 2021; Accepted: 9 March 2022;
Published online: 12 May 2022

References

1. Dietrich, T. et al. New constraints on the supranuclear equation of state and the Hubble constant from nuclear physics—multi-messenger astronomy. *Science* **370**, 1450–1453 (2020).
2. Aartsen, M. G. et al. Multimessenger observations of a flaring blazar coincident with high-energy neutrino IceCube-170922A. *Science* **361**, eaat1378 (2018).
3. Abbott, B. P. et al. Observation of gravitational waves from a binary black hole merger. *Phys. Rev. Lett.* **116**, 061102 (2016).
4. Abbott, B. et al. GW170817: observation of gravitational waves from a binary neutron star inspiral. *Phys. Rev. Lett.* **119**, 161101 (2017).
5. Harry, G. M. et al. Advanced LIGO: the next generation of gravitational wave detectors. *Class. Quantum Gravity* **27**, 084006 (2010).
6. Aasi, J. et al. Advanced LIGO. *Class. Quantum Gravity* **32**, 074001 (2015).
7. Acernese, F. et al. Advanced Virgo. *Class. Quantum Gravity* **32**, 024001 (2015).
8. Abbott, B. P. et al. Prospects for observing and localizing gravitational-wave transients with Advanced LIGO, Advanced Virgo and KAGRA. *Living Rev. Relativ.* **23**, 3 (2020).
9. Usman, S. A. et al. The PyCBC search for gravitational waves from compact binary coalescence. *Class. Quantum Gravity* **33**, 215004 (2016).
10. Cannon, K. et al. GstLAL: a software framework for gravitational wave discovery. Preprint at https://arxiv.org/abs/2010.05082 (2020).
11. Abbott, B. et al. Search for subsolar mass ultracompact binaries in Advanced LIGO's second observing run. *Phys. Rev. Lett.* **123**, 161102 (2019).
12. Aso, Y. et al. Interferometer design of the KAGRA gravitational wave detector. *Phys. Rev. D* **88**, 043007 (2013).
13. Somiya, K. Detector configuration of KAGRA—the Japanese cryogenic gravitational-wave detector. *Class. Quantum Gravity* **29**, 124007 (2012).
14. Zevin, M. et al. Gravity Spy: integrating advanced LIGO detector characterization, machine learning, and citizen science. *Class. Quantum Gravity* **34**, 064003 (2017).
15. Davis, D. et al. LIGO detector characterization in the second and third observing runs. *Class. Quantum Gravity* **38**, 135014 (2021).
16. Cuoco, E. et al. Enhancing gravitational-wave science with machine learning. *Mach. Learn. Sci. Technol.* **2**, 011002 (2021).
17. Vajente, G. et al. Machine-learning nonstationary noise out of gravitational-wave detectors. *Phys. Rev. D* **101**, 042003 (2020).
18. Ormiston, R., Nguyen, T., Coughlin, M., Adhikari, R. X. & Katsavounidis, E. Noise reduction in gravitational-wave data via deep learning. *Phys. Rev. Res.* **2**, 033066 (2020).
19. George, D. & Huerta, E. A. Deep learning for real-time gravitational wave detection and parameter estimation: results with Advanced LIGO data. *Phys. Lett. B* **778**, 64–70 (2018).
20. Schäfer, M. B. & Nitz, A. H. From one to many: A deep learning coincident gravitational-wave search. *Phys. Rev. D* **105**, 043003 (2022).
21. Green, S. R., Simpson, C. & Gair, J. Gravitational-wave parameter estimation with autoregressive neural network flows. *Phys. Rev. D* **102**, 104057 (2020).
22. Gabbard, H., Messenger, C., Heng, I.S. et al. Bayesian parameter estimation using conditional variational autoencoders for gravitational-wave astronomy. *Nat. Phys.* **18**, 112–117 (2022).
23. Mukund, N., Abraham, S., Kandhasamy, S., Mitra, S. & Philip, N. S. Transient classification in LIGO data using difference boosting neural network. *Phys. Rev. D* **95**, 104059 (2017).
24. Mittal, S. & Vetter, J. S. A survey of CPU–GPU heterogeneous computing techniques. *ACM Comput. Surv.* **47**, 69 (2015).
25. Duarte, J. et al. FPGA-accelerated machine learning inference as a service for particle physics computing. *Comput. Softw. Big Sci.* **3**, 13 (2019).
26. Caulfield, A. et al. A cloud-scale acceleration architecture. In *Proc. 49th Annual IEEE/ACM International Symposium on Microarchitecture* (IEEE, 2016).
27. Sfiligoi, I. et al. Expanding IceCube GPU computing into the Clouds. In *2021 IEEE 17th International Conference on eScience (eScience)* 227–228 (IEEE, 2021).
28. Wang, M. et al. GPU-Accelerated Machine Learning Inference as a Service for Computing in Neutrino Experiments. *Front. Big Data* **3** (2021).
29. Krupa, J. et al. GPU coprocessors as a service for deep learning inference in high energy physics. *Mach. Learn. Sci. Technol.* **2**, 035005 (2021).
30. Rankin, D. et al. FPGAs-as-a-Service Toolkit (FaaST). In *2020 IEEE/ACM International Workshop on Heterogeneous High-performance Reconfigurable Computing (H2RC)* 38–47 (IEEE, 2020).
31. Abbott, B. P. et al. Characterization of transient noise in Advanced LIGO relevant to gravitational wave signal GW150914. *Class. Quantum Gravity* **33**, 134001 (2016).
32. Triton Inference Server v.2.12.0 https://developer.nvidia.com/nvidia-triton-inference-server (NVIDIA, 2021).
33. Abadie, J. et al. First low-latency LIGO+Virgo search for binary inspirals and their electromagnetic counterparts. *Astron. Astrophys.* **541**, A155 (2012).
34. Nitz, A. H., Dal Canton, T., Davis, D. & Reyes, S. Rapid detection of gravitational waves from compact binary mergers with PyCBC Live. *Phys. Rev. D* **98**, 024050 (2018).
35. Abbott, R. et al. Search for gravitational waves associated with gamma-ray bursts detected by Fermi and Swift during the LIGO–Virgo run O3a. *Astrophys. J.* **915**, 86 (2021).
36. Usman, S. A. et al. The PyCBC search for gravitational waves from compact binary coalescence. *Class. Quantum Gravity* **33**, 215004 (2016).
37. Abbott, R. et al. GWTC-2: compact binary coalescences observed by LIGO and Virgo during the first half of the third observing run. *Phys. Rev. X* **11**, 021053 (2021).
38. Holzman, B. et al. HEPCloud, a new paradigm for HEP facilities: CMS Amazon Web Services investigation. *Comput. Softw. Big Sci.* **1**, 1 (2017).

39. Reitze, D. et al. Cosmic Explorer: The U.S. Contribution to Gravitational-Wave Astronomy beyond LIGO. *Bull. Am. Astron. Soc.* **51** (2019).
40. Kurtzer, G. M. et al. hpcng/singularity: singularity 3.7.3. https://zenodo.org/record/1310023 (2021).

Acknowledgements

We are grateful for computational resources provided by the LIGO Laboratory at Caltech, Livingston, LA, and Hanford, WA. The LIGO Laboratory has been supported under National Science Foundation (NSF) grants PHY-0757058 and PHY-0823459. A.G., D.R., T.N., P.H. and E.K. are supported by NSF grants 1934700 and 1931469, and D.R. additionally by the IRIS-HEP grant 1836650. J.K. is supported by NSF grant 190444. M.S. and M.C. are supported by NSF grant PHY-2010970. Work supported by the Fermi National Accelerator Laboratory, managed and operated by Fermi Research Alliance, LLC under contract DE-AC02-07CH11359 with the US Department of Energy. The US Government retains and the publisher, by accepting the article for publication, acknowledges that the US Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for US Government purposes. Cloud credits for this study were provided by the Internet2-managed Exploring Clouds for Acceleration of Science (NSF grant PHY-190444). Additionally we would like to thank the NSF Institute for AI and Fundamental Interactions (cooperative agreement PHY-2019786). We are also grateful for the support provided by S. Anderson in the realization and testing of our workflow within the LDG. Finally, we thank A. Pace for providing useful comments on the manuscript.

Author contributions

A.G. and D.R. are the primary authors of the manuscript. J.K. integrated applications in HEPCloud. S.T. and B.H. support and operate HEPCloud. M.S., M.C., E.K. and T.N. support development of DeepClean. All authors contributed to editing of the manuscript.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s41550-022-01651-w.

Correspondence should be addressed to Alec Gunny or Dylan Rankin.

Peer review information *Nature Astronomy* thanks Alexander Nitz and the other, anonymous, reviewer(s) for their contribution to the peer review of this work.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.