Research

Portable Acceleration of CMS Computing Workflows with Coprocessors as a Service

The CMS Collaboration[*]

CERN,Geneva, Switzerland

Received: 23 February 2024 / Accepted: 23 July 2024

Abstract Computing demands for large scientific experiments, such as the CMS experiment at the CERN LHC, will increase dramatically in the next decades. To complement the future performance increases of software running on central processing units (CPUs), explorations of coprocessor usage in data processing hold great potential and interest. Coprocessors are a class of computer processors that supplement CPUs, often improving the execution of certain functions due to architectural design choices. We explore the approach of Services for Optimized Network Inference on Coprocessors (SONIC) and study the deployment of this as-a-service approach in large-scale data processing. In the studies, we take a data processing workflow of the CMS experiment and run the main workflow on CPUs, while offloading several machine learning (ML) inference tasks onto either remote or local coprocessors, specifically graphics processing units (GPUs). With experiments performed at Google Cloud, the Purdue Tier-2 computing center, and combinations of the two, we demonstrate the acceleration of these ML algorithms individually on coprocessors and the corresponding throughput improvement for the entire workflow. This approach can be easily generalized to different types of coprocessors and deployed on local CPUs without decreasing the throughput performance. We emphasize that the SONIC approach enables high coprocessor usage and enables the portability to run workflows on different types of coprocessors.

Keywords CMS · Offline and computing · Machine learning

1 Introduction

During the first two runs of the CERN LHC [1], the ATLAS [2] and CMS [3] Collaborations have analyzed trillions of high-energy proton–proton or lead–lead collisions and produced an extensive suite of physics results. Among these are the discovery of the Higgs boson [4–6] in the standard model (SM) and stringent constraints on various beyond the SM physics scenarios, such as supersymmetry [7–14] and exotic heavy-particle or dark matter candidate production [15–21]. In order to measure the SM with higher precision and search for new processes with lower cross-sections, the amount of data that is delivered by the LHC and processed by the experiments is expected to increase dramatically in ongoing and future physics runs [22,23].

The high data-taking rates and increasing event complexity of the ATLAS and CMS experiments present a significant computational challenge for data processing [24,25]. A two-level trigger system is employed to run fast algorithms and reduce the data rate from 40 TB/s to about 10 GB/s [26–28]. While this is a significantly smaller rate, it is still very challenging for subsequent processing steps. As discussed in Refs. [29–31], even with optimistic expectations for computing research and development, the projected computing needs for CMS will be only narrowly satisfied.

At present, data processing is mainly carried out using central processing units (CPUs) but their expected performance increase is limited [32]. Nevertheless, data processing can be supported with a variety of modern architectures, such as graphics processing units (GPUs), field-programmable gate arrays (FPGAs), application-specific integrated circuits (ASICs), or Graphcore intelligence processing units (IPUs) [33,34], which can collectively be referred to as coprocessors. These architectures are becoming increasingly popular because of their large numbers of processing units, inherent parallelization designs, and more energy-efficient and environmentally friendly computing, especially suited for machine learning (ML) algorithm computations.

Within high-energy physics (HEP), deep-learning (DL) algorithms are already widely used for regression and classification tasks, and their popularity is growing rapidly [35–39]. Because of this growth, inference execution for these algorithms consumes increasingly large fractions of the overall processing load. However, the processing load can be shared and accelerated by using heterogeneous computing

[*] e-mail: cms-publication-committee-chair@cern.ch (corresponding author)

Published online: 04 September 2024

architectures. Therefore, developing a framework to enable and optimize the deployment and portability of coprocessors is of considerable interest to HEP experiments [40,41]. As a proof of concept, this paper focuses on accelerating the inference for ML algorithms in one of the CMS data processing stages, explained in "The CMS Data Tiers" section. These algorithms collectively take about 10% of the total processing time of that stage. For this approach to resolve the future computing challenge, ML algorithms would need to be used for larger portions of CMS data processing. Utilizing ML for tasks such as tracking [42–45], clustering [46–48], particle reconstruction [49–53], and particle identification [54–60] is an active area of research.

One of the approaches for the coprocessor deployment is to equip every CPU machine with coprocessors, referred to as directly connected. In this scenario, every CPU thread within the machine can communicate with the coprocessor. However, since the coprocessor-to-CPU ratio must be determined before deployment, the coprocessor resources are unlikely to be optimally utilized, leading to either unused resources when undersaturating, or decreased performance when oversaturating. In addition, it is difficult to utilize additional or more advanced coprocessor resources after deployment.

An alternative approach is inference as a service (IaaS), where coprocessor resources are separated from CPU machines. As illustrated in Fig. 1, in this scheme CPU-based *clients* can send the computing request with the necessary information to coprocessor-based *servers* via network calls. The servers running on coprocessor resources can perform computing tasks upon request. This removes the restriction of a coprocessor only being used by the CPUs directly connected to it, allowing it to accept processing requests from any CPUs (local or remote), as long as network communications are available. Certain types of coprocessors can be allocated for specific tasks, and the coprocessor-to-CPU ratio is flexible. Resource utilization can, therefore, be optimized based on specific tasks, as the number of client-side jobs using a single server can be varied depending on the computational demands of a given task. Furthermore, at the software level, since the support for coprocessors and CPU workflows is separated, it is easier to support different types of coprocessors. This ensures algorithm portability with minimal maintenance burden. The implementation of IaaS in experimental software frameworks can be accomplished using the Services for Optimized Network Inference on Coprocessors (SONIC) approach [61], as described in " The SONIC Approach" section.

The SONIC approach has previously been demonstrated using FPGAs [61,62] and GPUs [63–66] with a variety of ML algorithms. These studies demonstrated that offloading ML algorithms with the SONIC approach adds little additional computational overhead from the client–server communications and other operations. Therefore, a corresponding increase in the throughput has been observed when offloading algorithms to these faster coprocessors.

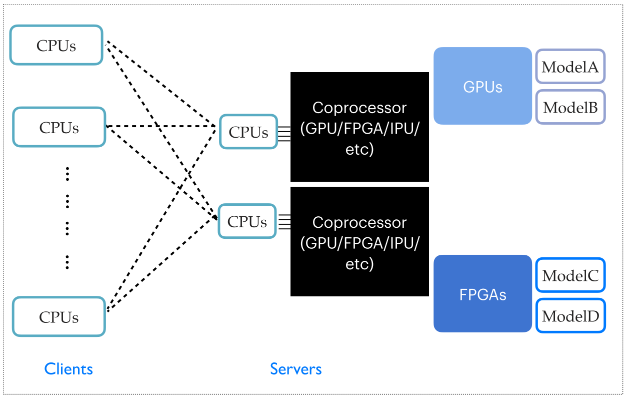


Fig. 1 An example inference as a service setup with multiple coprocessor servers. Clients usually run on CPUs, shown on the left side; servers hosting different models run on coprocessors, shown on the right side

Heterogeneous computing frameworks using GPUs have appeared recently in multiple non-IaaS contexts in HEP as well. One of the first significant real-time applications was in the ALICE high-level trigger (HLT) system [67], where GPUs were used to accelerate track reconstruction. Similarly, a fully GPU-based first-level trigger has been implemented in LHCb [68], which runs on 200 GPUs [69]. In CMS, a GPU-specific version of the pixel tracking domain (non-ML) algorithm called PATATRACK was developed, along with several other reconstruction algorithms, which are now employed in the HLT for Run 3 to reduce the processing time per event [70,71]. Further review of GPU usage in real-time applications for HEP can be found in Ref. [72].

In this paper, we take one data production workflow, called the Mini-AOD production workflow [73], as an example, and study the performance gains of applying the IaaS framework to this workflow. The abbreviation "AOD" comes from a lower-level data format called Analysis Object Data (AOD) discussed in The CMS Detector, Software, and Computing" section. In the current Mini-AOD production workflow, which is a data refinement and slimming step, about 10% of the computing time is consumed by ML algorithm inference, which can be easily accelerated on GPUs. We first summarize studies of the optimization and acceleration of the inference of each ML algorithm on GPUs. Then we show that the IaaS scheme, which is implemented in the CMS software framework CMSSW [74,75] via the SONIC approach, not only decreases processing time but can also be applied in large-scale production to optimize GPU utilization. Finally, we show that the SONIC approach can be easily ported to different types of coprocessors such as IPUs, and it can also run on local CPUs without decreasing the throughput. The study of power, power savings, and sustainability is also important, but is beyond the scope of this paper.

The paper is organized as follows. "The CMS Detector, Software, and Computing" section provides a brief overview of the CMS computing architecture and different data tiers for production. "The SONIC Approach" section discusses the SONIC approach in detail, including the technical implementation in CMSSW, the current inference servers, and features of the approach. " Physics Data Sets, Algorithms, and Benchmark Setup" section includes the data set used for the studies and the algorithms that can currently use the SONIC approach for inference. "Performance" and "Portability"sections provide detailed studies evaluating the computational performance of this approach in the Mini-AOD production workflow. Finally, Summary summarizes the studies and discusses future plans.

2 The CMS Detector, Software, And Computing

2.1 Introduction to the CMS Experiment

The LHC provides countercirculating beams of high-energy protons or heavy ions, such that bunches of particles in these beams can interact with each other in the center of the CMS detector [3] nearly every 25 ns. When particles from the countercirculating beams collide, a large variety of physical processes can occur, which lead to the creation of either fundamental or composite particles. These particles, or their decay products, can then propagate into the CMS detector, which is designed to measure their energy and momentum. In the context of this paper, we will refer to a readout cycle of the detector as an event. The detector itself comprises multiple layers including silicon pixels and strips, crystal electromagnetic calorimeters, sampling hadron calorimeters, and muon spectrometers. Each component of the detector has active elements that create electrical signals when particles interact with the detector. Each discrete signal is called a hit.

Events of interest are selected using a two-tiered trigger system. The first level (L1), composed of custom hardware processors, uses information from the calorimeters and muon detectors to select events at a rate of around 100 kHz within a fixed latency of 4μs [76]. The second level, known as the HLT, consists of a farm of processors running a version of the full event reconstruction software optimized for fast processing, and reduces the event rate to around 1 kHz before data storage [28]. Processing in the trigger system is referred to as online computing, while subsequent processing is known as offline computing.

2.2 The CMSSW Framework

The CMSSW framework is an open-source software framework that is used in triggering, data formatting and processing, simulation, and offline analysis [74,75]. Components of the framework are used in combination to extract high-level physics information for each event from detector hits. The CMSSW framework processes each event with a sequence of algorithms, converting hits from electrical signals to position and energy measurements, linking these measurements into clusters [77] and trajectories [78], combining trajectories and clusters into single-particle representations or jets corresponding to hadronic showers [79]. Additional algorithms in CMSSW, e.g., ML-based reconstruction algorithms, can be run to determine quantities such as an overall imbalance of the momentum in the direction perpendicular to the beam or to tag jets as containing or being produced by certain particles.

The CMSSW framework uses Intel Threading Building Blocks [80] to enable task-based multithreading. As explained in Ref. [81], this multithreading implementation allows for asynchronous nonblocking calls to external resources, such as a GPU, via ExternalWork. This setup optimizes CPU resource utilization by minimizing downtime; the CPU is allowed to continue executing algorithms that do not require a coprocessor or depend on the results of the coprocessor-dependent algorithm while waiting for the external call to return.

2.3 The CMS Data Tiers

The CMSSW framework is used both in online and offline contexts within CMS. While the L1 trigger is hardware-based, the HLT is composed of algorithms in CMSSW. The framework also contains the algorithms that are used to process the raw data after they have been stored, deriving the higher-level information useful for a wide range of data analyses based on reconstructed objects. The centralized CMS offline data processing flow involves three steps, which are performed both for raw data and simulated data sets. In the first step, an AOD format for every event is derived. This contains high-level information, such as reconstructed particles and jets, but the data size is large. In the second step, a slimmed, higher level Mini-AOD derivation is created [73]. Mini-AOD files are designed to be relatively small and accessible, serving as an intermediate step and the standard foundation for a variety of CMS physics analyses. Finally, in the third step, a further slimmed Nano-AOD format is created that contains only very high-level physics observables [82]. This format is commonly used directly for physics analyses. Data sets are reprocessed regularly to incorporate the latest Monte Carlo event generator tunes [83], calibrations, and algorithm improvements. Table 1 summarizes the average event size of these different data tiers with 2016–2018 (Run 2) data-taking conditions [84].

In the scope of the studies in this paper, we choose Mini-AOD production as our test case. Mini-AOD files are derived from the AOD data format, reducing the size per event by an order of magnitude. Mini-AOD processing involves a wide

Table 1 Average event size of different CMS data tiers with Run 2 data-taking conditions [73,82,85]

Data tier	Event size [kB/event]
Raw	1000
AOD	480
Mini-AOD	35–60
Nano-AOD	1–2

variety of algorithms that propagate, filter, and reanalyze the AOD input objects.

3 The SONIC Approach

This section describes the implementation of the SONIC approach in CMSSW and the server technology currently used, which is the NVIDIA Triton Inference Server (Triton) [86]. The benefits of running inference with the SONIC approach along with additional complexity and other implications are also discussed in this section.

3.1 Implementation in CMSSW

The SONIC approach is implemented in CMSSW through the ExternalWork framework component [81] and accesses coprocessor resources on remote servers via gRPC Remote Procedure Calls (gRPCs), which is a cross-platform open-source high-performance remote procedure call framework originally developed by Google [87]. An illustration of this procedure, where client jobs make asynchronous, nonblocking gRPC calls to remote servers, is shown in Fig. 2. Multiple servers can run on multiple coprocessors, with load balancers in between. Asynchronous communication allows client CPUs to process other tasks in parallel while data are transferred between the client and the server and the inference task is processed on the server. An important aspect of this scheme is that the client-side code does not need to be able to run any particular inference packages or frameworks; it simply has to collect the relevant input data for a trained model, communicate that information to the server in the expected format, and handle the output from the server.

The client-side framework code for the SONIC approach is split into two packages: a core package [89] containing class templates and other common infrastructure for the IaaS approach, and a dedicated package [90] to interact specifically with the Triton server. SONIC modules provide a similar interface to standard CMSSW modules. The partitioning into two packages reflects that the SONIC approach can be implemented for multiple server backends. For example, the first implementation [61,91] used the Microsoft Brainwave service [92], which provides FPGA resources. Another potential backend service is the TensorFlow as a Service framework, which can provide access to TensorFlow-based ML models via the HTTP protocol [93]. This framework was introduced along with the Machine Learning as a Service pipeline for HEP (MLaaS4HEP), which is a streamlined mechanism for training and deploying models from data in ROOT data format files [94]. In practice, given the generality and openness of the protocols used by Triton, which are themselves an extension of the KServe standard [95], we expect that future development of the SONIC approach will continue to use these protocols, even if the backends or coprocessors change. Further discussion of different backends and coprocessors can be found in “Portability“ section.

A central TritonService is provided to keep track of all available servers and which models each of them serves. By default, each client-side module only needs to specify the model it needs, and the TritonService will automatically find a server hosting that model. A client-side module can optionally specify a preferred server, which the TritonService will use if the server is found and confirmed to serve the required model.

Within CMSSW, a mechanism has been implemented to account for the possibility that a client job cannot access a specified server for whatever reason. In this case, a “fallback” server is automatically created using either local GPU resources if they are available or the CPU resources allocated to the client job in question. The client then makes gRPC calls to that local fallback server, which introduces negligible latency. Detailed studies related to these fallback servers are discussed later in “The CPU Fallback Server“ section. In general, the server overhead consumes very little of the CPU resources beyond what would be used for conventional inference, such that the per-event processing time is not strongly affected by the SONIC approach relative to running without it. Fallback servers are automatically shut down when the job finishes.

3.2 The NVIDIA Triton Inference Server

As shown in Fig. 2, the server-side implementation of the SONIC approach in CMSSW currently uses Triton for inference on coprocessors [86,96]. Triton is an open-source solution whose protocols are public and extensible, as noted above. It supports inference of ML algorithms, called models, in most modern formats, including PyTorch [97], TensorRT (TRT) [98], ONNX Runtime (ONNX) [99], TensorFlow [100], and XGBoost [101]. It also supports custom backends for alternative tasks, such as classical rule-based domain algorithms and inference on different types of coprocessors. Several features of Triton are worth highlighting:

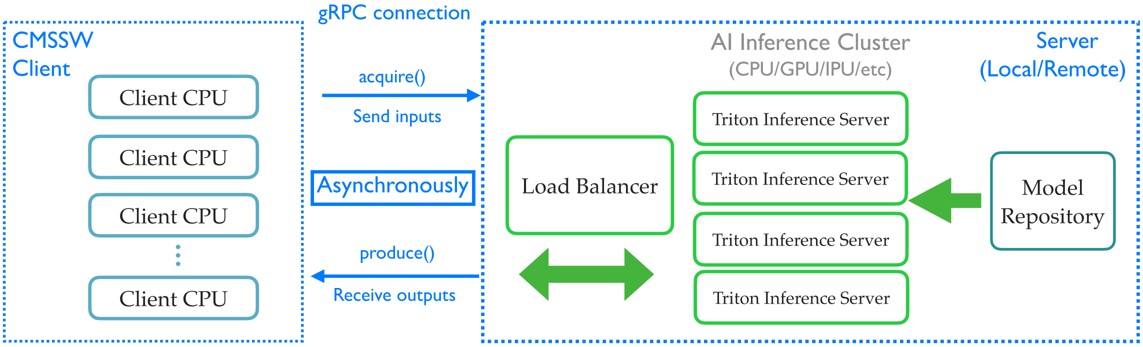


Fig. 2 The SONIC implementation of IaaS in CMSSW. The figure also shows the possibility of an additional load-balancing layer in the SONIC scheme. For example, if multiple coprocessor-enabled machines are used to host servers, a Kubernetes engine can be set up to distribute inference calls across the machines [88]. Image adapted from Ref. [64]

- *Multiple model instances*: A single server can host multiple models at the same time or even multiple instances of the same model to allow concurrent inference requests.
- *Dynamic batching*: Usually coprocessors have many processing units and the number of operations from one inference call is not enough to fully utilize the coprocessors. With dynamic batching, if multiple inference calls are made within a window of time, the server can concatenate the inputs of all calls into a single batch, which improves the GPU utilization and therefore increases the overall throughput. In practice, the time window is typically chosen to be about the client-side CPU processing time per event, such that the inference can be processed in time without delay.
- *Model analyzer*: Parameters like the number of model instances on a single server, the length of a batching window, and the optimal batch size are tunable and can be optimized on a case-by-case basis. TRITON provides a model analyzer tool to aid in this optimization [102]. It can mimic the clients and send randomized inputs or some pre-saved data to the server. The server performance is measured, and the optimal deployment configuration can be determined by scanning the parameter space.
- *Ragged batching*: Traditionally, inference can be performed on a batch of multiple inputs as long as each input is of the same size. In HEP data, the size of the input for inference can vary from one instance to another, making it harder to batch them together. For example, if an algorithm uses information from every particle in an event as input, it is difficult to batch inference requests from multiple events because events can have a wide range of numbers of particles. Ragged batching allows inference requests with different sizes to be batched together, thereby improving the performance. This feature is relatively new and not yet fully studied in this paper.

TRITON servers can use one or multiple GPUs on the same machine with a built-in load balancer. They can also run purely on CPU resources when there are no GPU resources available. For other types of coprocessors, TRITON servers can also be used with the help of custom backends. A server requires a trained model file and a configuration file specifying input and output variable names, shapes, types, and model versions, along with the preferred batch size and other details that can be acquired through the inference optimizations. These model and configuration files are currently accessible through the CernVM-File System (CVMFS) [103], and are tracked by the CMSSW release management system.

3.3 Advantages of the SONIC Approach

Inference as a service, as implemented in the SONIC approach, provides several advantages and benefits, which are summarized here. Many of these features arise from the differences between the IaaS approach and the more traditional approach of HEP software frameworks to use only local computing resources. These features include:

- *Containerization*: The SONIC approach factorizes ML frameworks out of the client software stack, i.e., CMSSW, reducing the workload to support a wide variety of ML models. With the SONIC approach, one can use any framework supported by TRITON, including custom backends, without needing to modify the CMSSW software stack to resolve library dependencies and ensure compatibility between all external packages. This allows us to pick the best inference backend for one algorithm, with less concern for the implementation details.
- *Simplicity*: Because of the containerization discussed above, SONIC client-side code is simpler and more general than the corresponding direct inference code. SONIC modules need only implement the conversion of input

data into the server's desired format and the reverse operation for output data.
- *Flexibility*: In the SONIC paradigm, the connections between client CPUs and server coprocessors are not fixed. The servers can be physically located nearby or far from clients. Clients from many machines can access a single server running on either one or multiple coprocessors. Similarly, a single client can access multiple different servers running on multiple different machines.
- *Efficiency*: The SONIC approach enables balanced utilization of coprocessor resources. By optimizing the coprocessor-to-CPU ratio for different tasks, it is easier to fully utilize coprocessor resources without oversaturating them.
- *Portability*: Through the use of the SONIC approach, client-side code does not have to be modified to take advantage of different types of coprocessors. Only a consistent protocol for communicating with the inference server is required, regardless of the underlying hardware: CPU, GPU, FPGA, IPU, or any other architecture.
- *Accessibility*: If GPUs or other coprocessors are not available locally, the only way to accelerate workflows is to access those resources remotely, as a service. The SONIC architecture implements this use case for CMS, allowing the use of remote coprocessors.

3.4 Limitations and Production Requirements

However, with these advantages come additional complexity and changes in resource usage. Production jobs using the SONIC approach rely on a separate server running different software, compared to the existing scheme in which jobs only execute CMSSW on local hardware. This implies several additional considerations:

- *Server failures*: Inference servers, remote or local, may experience software or hardware failures that prevent them from running. These new failure modes are mostly independent from existing known sources of failures, potentially leading to an overall increase in the rate of job failures. However, these failures can be mitigated with server-side technology, such as the load balancer Kubernetes [88], and client-side protocols, like the local fallback server.
- *Network usage*: The use of remote inference servers necessarily implies an increase in network traffic, as input and output data must be communicated over the network. Typically, input data are much larger than output data; the total usage depends on the algorithm. For the Mini-AOD production workflow tests presented here, the network usage is discussed in "Large-Scale Tests" section. High network usage can be mitigated using compression, with some tradeoff in throughput from the additional operations to compress and decompress the data. In the studies done here, we have not observed significant issues with network usage, and as a result, analysis of the tradeoff between compression and network usage is not included.
- *Memory usage*: The use of remote inference servers reduces the local memory usage of production jobs, compared to the direct inference approach. However, the use of local fallback servers, whether to mitigate remote server failures or take advantage of the containerization and portability of the SONIC approach, implies increased memory usage. Running the server process locally is generally expected to use more memory than the corresponding direct inference libraries. Measurements of memory usage are presented in "The CPU Fallback Server" section.

These potential drawbacks, especially uncorrelated failures and network usage, are similar to those from other distributed services used in CMS production, such as the conditions database or XROOTD [104,105]. These can potentially impact the processing performance and should be studied more intensively.

Handling this additional complexity requires new components to be deployed in the CMS workflow management system. We provide below several examples of such operational concerns.

- *Server discovery*: The SONIC approach allows the use of remote coprocessor resources, which requires the information of servers running on these resources, e.g., IP addresses, port numbers, served models, and number of GPUs, etc. to be available to the client jobs. Such information can be collected in the site configurations and provided by the job submission system or a central service. For the studies presented in this paper, servers are launched manually, and their addresses are written into the CMSSW configuration files.
- *Load balancing*: Different production jobs running on different data sets have different coprocessor resource demands. A load balancer, such as Kubernetes, can be set up to dynamically load and unload models on different servers and distribute client inference requests to these servers. The load balancing for large-scale inference requests is tested and discussed in "Large-Scale Tests" section.
- *Versioning*: CMS production requires the versions of CMSSW, servers, ML models, and ML backends to be controlled and recorded for proper provenance tracking. The ML model versions are already tracked by the CMSSW release management system; the server and backend versioning should be included in the same system for consistency. For the studies presented in this paper, the prebuilt TRITON server provided by NVIDIA was used.

- *Optimization procedure*: While the SONIC approach enables the benefits described in "Advantages of the SONIC Approach" section, it does not necessarily make them trivially attainable. Some use-case-specific optimizations are required before large-scale deployment, as explained in more detail in "Performance" section. For example, each model should be analyzed individually to find a preferred batch size. Similarly, while the flexible coprocessor-to-CPU ratio permits more efficient utilization of resources, one must first determine an appropriate coprocessor-to-CPU ratio. This is further complicated by the fact that the ratio can depend on the hardware used, such as CPU or GPU type, and on the physics content of the data set being processed.

4 Physics Data Sets, Algorithms, and Benchmark Setup

This section describes the physics data set, the ML algorithms in CMSSW that can currently use the SONIC approach for inference, and the computing resources used in the studies.

4.1 Data Set and Software Versions

In the studies, we chose to process a simulated Run 2 data set of events with one top quark and one anti-top quark ($t\bar{t}$), as it includes many types of physics objects, including leptons, heavy-flavor jets, and missing transverse momentum. Here, a heavy-flavor jet is one that originates from a charm or bottom quark, and missing transverse momentum, or $\vec{p}_{\mathrm{T}}^{\mathrm{miss}}$, is defined as the negative vector sum of the transverse momenta of all of the reconstructed particles in an event, and its magnitude is denoted as $p_{\mathrm{T}}^{\mathrm{miss}}$ [106]. The data set was copied to local disk to factor out the effects of remote input–output (I/O) limitations for the benchmarks. The CMSSW version CMSSW_12_0_0_pre5 and Triton server version 21.06 were used for the studies.

4.2 Algorithms Supported by the SONIC Approach

Adapting an ML algorithm to work with the SONIC approach requires a small effort to write code to prepare the network inputs and save the network outputs within CMSSW. A comparison of the producers used for direct inference and for the SONIC approach can be found in Refs. [107] and [108], respectively. Most of the pre- and post-processing steps are the same, while the operations that send inference requests to the ML backends directly or to the servers are different.

In these studies, we tested three independent and computing-intensive ML-based algorithms in the Mini-AOD workflow. These algorithms were chosen to illustrate the performance of the SONIC approach for models with differing input and output sizes, physics applications, and backends, as detailed below.

4.2.1 The ParticleNet Algorithm

Graph neural networks (GNNs) have been demonstrated to achieve state-of-the-art performance for identifying jets as arising from specific particles, a task known as jet tagging [54–56]. ParticleNet [55] is a GNN-based algorithm for jet tagging and regression that represents jets as "particle clouds." This algorithm was trained in PyTorch [97] and exported to the ONNX format. With the SONIC approach, it is possible to perform inference with ParticleNet in the following formats: ONNX; PyTorch; and PyTorch with TRT [98], which optimizes model performance on NVIDIA devices.

There are four different trained versions of ParticleNet currently running in the Mini-AOD workflow for different purposes:

1. tagging anti-k_{T} jets [109], clustered with the FastJet package [110], with a radius parameter of 0.4 (AK4 jets), abbreviated PN-AK4 [111],
2. tagging anti-k_{T} jets with a radius parameter of 0.8 (AK8 jets) [112],
3. mass-decorrelated tagging for AK8 jets [112], and
4. mass regression for AK8 jets [113].

The three AK8 ParticleNet algorithms are abbreviated PN-AK8 in the following sections. All ParticleNet variations can be hosted on Triton servers.

The inputs to ParticleNet are the kinematic and flavor properties of the particle constituents of each jet and the secondary vertices associated with the jet, including 20 features for one particle and 11 features for one vertex. Up to 100 particles and 10 vertices are used in the inputs for AK8 jets; if there are more than 100 particles in a jet, the 100 particles with the highest transverse momenta are used, and if there are more than 10 vertices, then the 10 with the highest displacement from the beamline are used. For AK4 jets, the maximum numbers of particles and vertices are 50 and 5, respectively. For the three tagging versions of ParticleNet, the outputs of each inference are category probabilities for a variety of predefined jet categories, such as the presence of a Higgs boson or the presence of a top quark. For the mass regression, the output is a single value: the predicted jet mass.

In Mini-AOD processing, inference is performed separately for each jet in a given event, so the number of ParticleNet inferences depends on the specific physics processes and can vary substantially from event to event. When running the standard CMSSW version of ParticleNet, no inference batching is performed, so each inference is truly performed separately. In this context, each inference can have a variable number of inputs with no padding involved. When using the SONIC approach for ParticleNet, it is simplest to batch all jets in an event into a single inference request, such that input particle and secondary vertex information for each jet in an event is sent to the server in a single request. Because ragged batching is not yet fully implemented in the inference server, the different inference requests must have the same number of inputs per batch. Therefore, in subsequent performance studies, we adopt a maximally padded approach, where zeros are added to the vectors of particles and vertices for AK8 (AK4) jets such that they have consistent lengths of 100 (50) and 10 (5), respectively.

An illustration of the jet content of the Run 2 simulated $t\bar{t}$ data set is given in Fig. 3. The distributions of the number of jets per event and particles per jet are provided for both AK4 and AK8 jets. As shown in the figure, the number of jets can vary dramatically from event to event, indicating the importance of dynamic batching.

In the configuration where all AK4 jets are padded to 50 particles and 5 vertices and all AK8 jets are padded to 100 particles and 10 vertices, a single four-threaded job processing a Run 2 $t\bar{t}$ event will generate about 140 kB/s of server input for AK4 jets and about 10 kB/s of server input for each of the AK8 jet versions of ParticleNet. At a processing rate of about 4 events per second, with about 16 AK4 jets per event, this corresponds to about 2 kB of information per jet, which is consistent with the number of float inputs per jet.

4.2.2 The DeepMET Algorithm

DeepMET [114] is a TensorFlow-based deep neural network model that estimates the $\vec{p}_{\mathrm{T}}^{\mathrm{miss}}$ in an event. The vector $\vec{p}_{\mathrm{T}}^{\mathrm{miss}}$ is associated with either the production of neutrinos or potential beyond the SM particles that could propagate through the detector interacting only weakly. The inputs to DeepMET are 11 features from each particle [79] in an event, with zero-padding up to 4 500 particle candidates for a given event. This zero padding is used for both the standard version of DeepMET and its SONIC implementation. In both cases, only one inference is made per event. DeepMET outputs two values for each event: the $\vec{p}_{\mathrm{T}}^{\mathrm{miss}}$ components in the transverse plane. Because every event necessarily has the same number of inputs, dynamic batching is automatically available via the SONIC approach, aiding inference efficiency in cases where many client jobs make concurrent requests within a certain time window (each client thread will make a request about once per second). A single four-threaded job processing a Run 2 $t\bar{t}$ event will generate about 1.3 MB/s of server input for DeepMET.

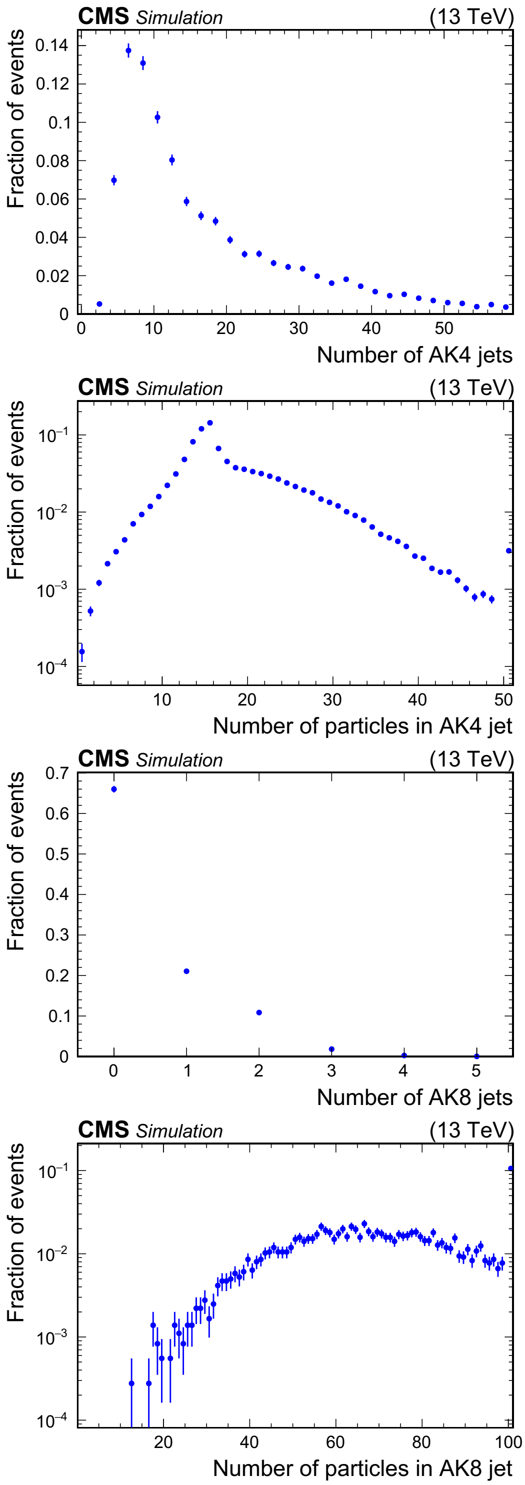


Fig. 3 The jet information in the Run 2 simulated $t\bar{t}$ data set used in subsequent studies. Distributions of the number of jets per event (left) and the number of particles per jet (right) are shown for AK4 jets (upper) and AK8 jets (lower). For the distributions of the number of particles, the rightmost bin is an overflow bin

4.2.3 *The DeepTau Algorithm*

The DeepTau algorithm [115] is a TensorFlow-based deep neural network model to identify hadronically decaying τ leptons from the jet collections. The inputs to the network include low-level particle features of electrons/photons, muons, and hadrons, and high-level features such as the τ lepton candidate kinematic information. The algorithm splits nearby cones into grid cells, and loops over the cells to collect the particle features. The low-level particle features are then processed by a convolutional neural network (CNN) to extract the particle-level information. The high-level features are processed by a fully connected neural network (FCNN) to extract the τ lepton candidate-level information. The outputs of the CNN and FCNN are combined and processed by a final FCNN to produce a four-dimensional vector that represents the probability that the candidate originates from a genuine τ lepton, a muon, an electron, or a quark or gluon jet. In the implementation of direct inference in CMSSW, the network is split into three sub-models. The first two networks process the lower-level input information separately, and the third network combines the information from the first two models with the high-level information and outputs the discriminator values. In the SONIC implementation, we use a combined model with zero-padded inputs, so that dynamic batching can be used and GPUs can be utilized more efficiently. A single four-threaded job processing a Run 2 $t\bar{t}$ event will generate about 4.7 MB/s of server input for DeepTau, making it the most demanding algorithm explored in this study in terms of server input.

4.3 Average Processing Time

The processing time is defined as the real-world time spent between starting and finishing processing one event. The processing time breakdown for the three algorithms highlighted above in the Run 2 $t\bar{t}$ events is presented in Table 2, measured with 1-thread jobs using one single CPU core. The per-event per-thread average processing time is about 1 s; PN-AK4 consumes about 4.3% of this time and PN-AK8 about 1.1%, while DeepMET and DeepTau take about 1.3 and 2.1%, respectively. Thus, for this collection of events, the algorithms supported by the SONIC approach account for about 9% of the total processing time. This fraction is dependent on the type of events. For example, if there are fewer jets per event, ParticleNet will consume a smaller fraction of the total processing time. While the fraction of the total workflow accelerated with the SONIC approach in this paper is less than 10%, an increasing number of ML algorithms are being integrated into CMS data processing. Because of this, it will be possible to accelerate a larger fraction of the total processing time with the SONIC approach in the future.

4.4 Computing Resources

Fermilab, via the LHC Physics Center, provides CPU-only batch resources and a set of interactive machines with NVIDIA Tesla T4 GPUs [116]. Through Fermilab, we were also able to steer the allocation of cloud resources (see below) using the HEPCloud [117] framework. These resources are physically located in Illinois.

The Google Cloud Platform (GCP) provides virtual machines (VMs) that are either CPU-only or enabled with NVIDIA Tesla T4 GPUs. By default, the CPUs are a mix of Skylake, Broadwell, Haswell, Sandy Bridge, and Ivy Bridge architectures [118]. In GCP, we created customized machines with specified numbers of CPU threads or different ratios of CPU threads to numbers of GPUs. Similarly, a customized, dynamic SLURM [119] cluster was created that could instantiate and deplete four-thread VMs on demand for medium-scale tests that ran jobs across $\mathcal{O}(1000)$ CPUs. The cluster's four-thread configuration was chosen so that four-threaded CMSSW jobs would saturate the node's resources, improving the reproducibility of timing tests. CPU-only VMs could also be instantiated through HEPCloud. In GCP, we also maintained GPU-enabled VMs running Triton servers that both the SLURM and HEPCloud client nodes could access. These resources are physically located in Iowa.

At the Purdue CMS Tier-2 computing cluster, tests were performed with reserved CPU-only and GPU-enabled machines. The CPU-only machines are 20-core Intel E5-2660 v3 machines, and the GPU-enabled machines each have two AMD EPYC-7702 CPUs [120] with an NVIDIA Tesla T4 GPU. Reserving these nodes allowed for controlled resource utilization, leading to more reproducible timing tests. These resources are physically located in Indiana.

The diversity of resource locations used in these studies demonstrates one of this approach's key features, namely that it enables the use of nonlocal resources. We were able to start a server in one location and have client jobs running at another site. One such study is presented in "Cross-site Tests“ section.

Within a GCP project, VMs do not have ingress bandwidth limits other than machine limits. These are above 10 GB/s, which, based on the scans in "Per-Algorithm Inference Optimization“ section and Table 2, is far above the amount of information that can be sent from client VMs to a server-hosting machine without saturating GPU resources. For example, in GCP, if a single GPU is used to host all model types, then in a typical running scenario, it can service about 33 simultaneous four-threaded client jobs, or 132 client cores. A single client core generates about 1.55 MB/s of traffic, so a total of 200 MB/s of ingress is expected per GPU at the saturation point, which is reached when the GPU is running at the maximum throughput and cannot handle any additional incoming requests. Similarly, the ingress bandwidth from the

Table 2 The average time of the Mini-AOD processing (without the SONIC approach) with one thread on one single CPU core.

Algorithm	Time [ms]	Fraction [%]	Input [MB]
PN-AK4	42	4.3	0.04
PN-AK8	11	1.2	0.01
DeepMET	13	1.3	0.33
DeepTau	21	2.1	1.18
PN-AK4+PN-AK8+DeepMET+DeepTau	88	8.9	1.55
Full workflow	990	100.0	—

The average processing times of the algorithms supported by the SONIC approach are listed in the column labeled "Time." The column labeled "Fraction" refers to the fraction of the full workflow's processing time that the algorithm in question consumes. Together, the algorithms currently supported by the SONIC approach consume about 9% of the total processing time. This table also contains the expected server input for each model type created per event in Run 2 $t\bar{t}$ events in the column labeled "Input"

server into the client VMs will be very small, as inference results are no more than $\mathcal{O}(10)$ float values per algorithm. The GCP does impose VM egress bandwidth limits, which is typically 0.25 GB/s per CPU for a VM to an internal IP address. However, given that a single client-side core will generate about 1.55 MB/s of network traffic, this is also well within the allowed limit. There are slightly more stringent restrictions for GCP bandwidth to external IP addresses. However, when we performed such tests as in "Cross-site Tests" section, typically only one external machine was involved, with levels of network traffic well within that allowed by the restrictions.

5 Performance

In this section, we discuss the performance of running the Mini-AOD production workflow with the SONIC approach. We compare to direct inference, which refers to the standard approach where the inference is performed using the ML backends integrated into CMSSW on CPUs.

As mentioned in " The NVIDIA Triton Inference Server" section, we first optimize the per-algorithm workflows to find the optimal configurations for the full production workflow. Next we check the impact of deploying servers on different sites. Finally, we mimic the real production jobs by running scale-up tests and evaluating the performance.

5.1 Per-Algorithm Inference Optimization

To maximize the resource efficiency and throughput benefits of the SONIC approach, we first perform single-model characterization studies independent of CMSSW. For example, to maximize GPU usage without oversaturation, we need to find the optimal ratio of client-side CPUs to server-side GPUs, batch size for inference in the TRITON server, and model configuration that will provide the highest throughput.

The latter two optimizations can be performed with the TRITON Model Analyzer tool [102]. This tool feeds inputs in the correct tensor format (either randomized numbers or real data) to a loaded model hosted on a server, allowing for robust characterization of processing time per inference or exploration of the impact of batch size. As an example, we measure the processing time and throughput of the ParticleNet algorithm for AK4 jet tagging on one NVIDIA Tesla T4 GPU, with different inference backends supported in TRITON: ONNX, ONNX with TRT, and PYTORCH [97] (labeled PT in the figures). The results are shown in Fig. 4.

For smaller batch sizes, the TRT version of the ParticleNet algorithm leads to the highest throughput in total inferences per second, while at higher batch sizes, the PYTORCH version gives higher throughput. In the version of ParticleNet supported by SONIC in CMSSW, all the jets in a single event are batched together, and there are on average 16 AK4 jets per event in our $t\bar{t}$ data set. To achieve higher batch sizes in a production scenario, multiple CMSSW clients would need to make an inference request to the same server within a relatively narrow time window. TRITON allows us to specify a preferred batch size, such that if many inference requests are queued within the time window, the server will perform inference with batches of approximately the specified size. For example, the peak throughput seems to plateau around a batch size of 100 for the PYTORCH version of ParticleNet.

Similar studies can be performed on other models as well. Figures 5, 6, and 7 show the processing time and throughput scans of PN-AK8 jet tagging, DeepMET, and DeepTau models, respectively. Some backend tests are skipped due to the model availability.

The model analyzer can also determine approximately how many inferences per second a single GPU can perform before saturation. For example, for the PYTORCH version of ParticleNet for AK4 jets, a single Tesla T4 GPU can perform about 5500 inferences per second without saturating. This corresponds to about 350 events per second, given the typical number of jets per event. Based on this, we can estimate how many CPU clients one GPU can support in parallel. A typical production configuration runs four-threaded Mini-AOD jobs,

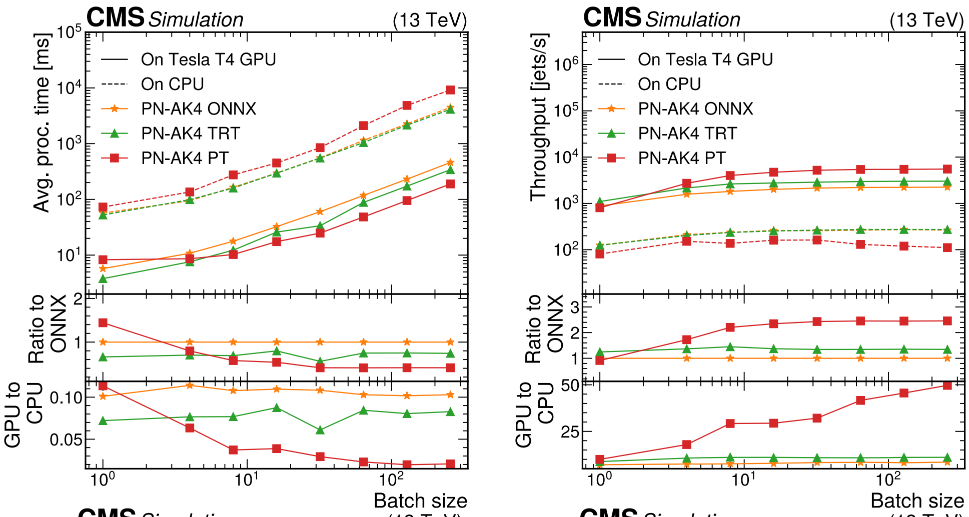


Fig. 4 Average processing time (left) and throughput (right) of the PN-AK4 algorithm served by a Triton server running on one NVIDIA Tesla T4 GPU, presented as a function of the batch size. Values are shown for different inference backends: ONNX (orange), ONNX with TRT (green), and PyTorch (red). Performance values for these backends when running on a CPU-based Triton server are given in dashed lines, with the same color-to-backend correspondence

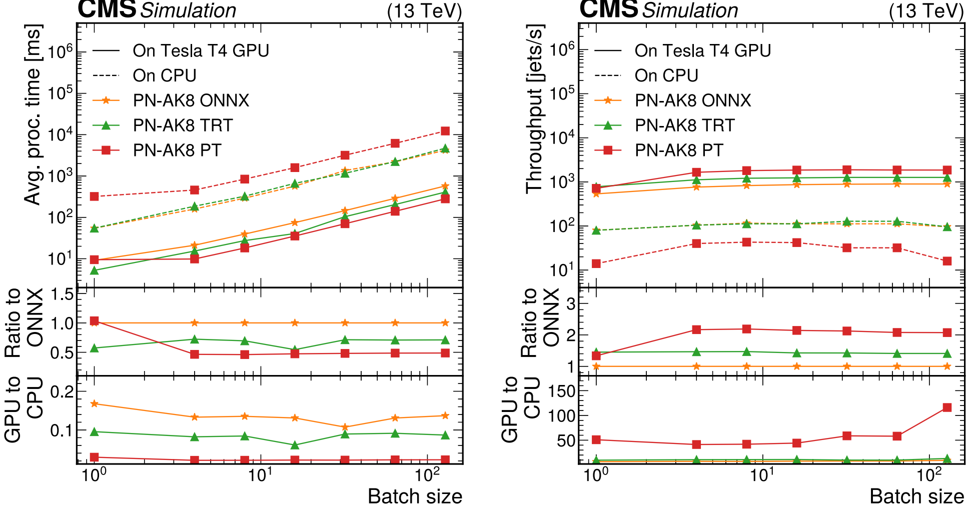


Fig. 5 Average processing time (left) and throughput (right) of one of the AK8 ParticleNet algorithms served by a Triton server running on one NVIDIA Tesla T4 GPU, presented as a function of the batch size. Values are shown for different inference backends: ONNX (orange), ONNX with TRT (green), and PyTorch (red). Performance values for these backends when running on a CPU-based Triton server are given in dashed lines, with the same color-to-backend correspondence

each of which processes about 3.9 events per second. Therefore, a single GPU should be able to handle about 90 four-threaded jobs in parallel running asynchronously, assuming it is being used exclusively for a single server hosting the PyTorch PN-AK4 model.

This expected saturation point can be tested directly by running the Mini-AOD workflow in cmssw and scanning the throughput as a function of the number of cmssw CPU clients pinging one GPU server. Figure 8 shows such tests for the PN-AK4, PN-AK8 jet tagging, DeepMET, and DeepTau models, which were performed in GCP using a custom SLURM cluster. For each model, a single server running on one NVIDIA T4 GPU was started on one cloud VM, and client-side jobs were executed in VMs that had 4 CPU threads. The tests for each model class were performed separately, and as one model class was being tested, the direct-inference versions of the other models were used.

The accelerated versions of the workflow can be compared with the dashed black line, which represents the average throughput of the workflow when the direct-inference versions of all the models were used. In this case, jobs were also started in the VMs with 4 CPU threads, but there was no communication with an external server. As there were no shared resources between jobs, there is no expected dependence on the number of synchronized jobs. The average processing time for this setup was determined very accurately by simply running a large number of jobs, so no associated error is shown in Fig. 8.

When offloading PN-AK4 inference to the GPU, we expect an improvement in the overall throughput of about 4% compared with direct inference, corresponding to the fraction of time taken by the total PN-AK4 processing shown in Table 2. Such an improvement is observed before saturation, where the throughput is stable as a function of the number of simultaneous CPU clients. The throughput decreases as the GPU starts to saturate because individual client-side jobs have to wait longer for inference requests to complete and return. The throughput becomes lower than CPU-only inference slightly above 160 four-threaded parallel jobs, i.e.,, equivalent to 640 single-threaded jobs, comparable with the expected saturation point from the model analyzer.

Similar analyses were performed for the other models. When all three variants of ParticleNet for AK8 jets are hosted on a single GPU, that GPU can serve about 190 simultaneous

Fig. 6 Average processing time (left) and throughput (right) of the DeepMET algorithm served by a TRITON server running on one NVIDIA Tesla T4 GPU, presented as a function of the batch size. Similar performance when running on a CPU-based TRITON server is given in dashed lines

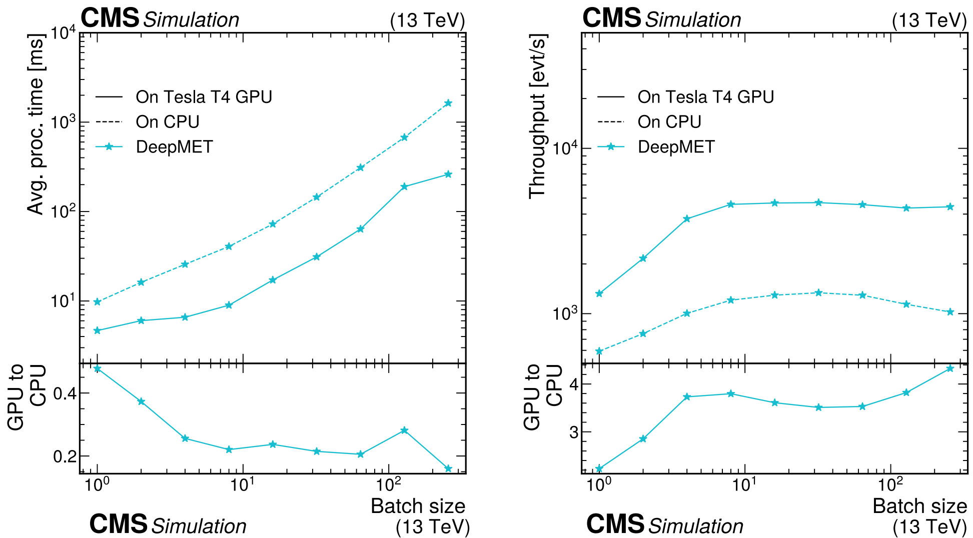
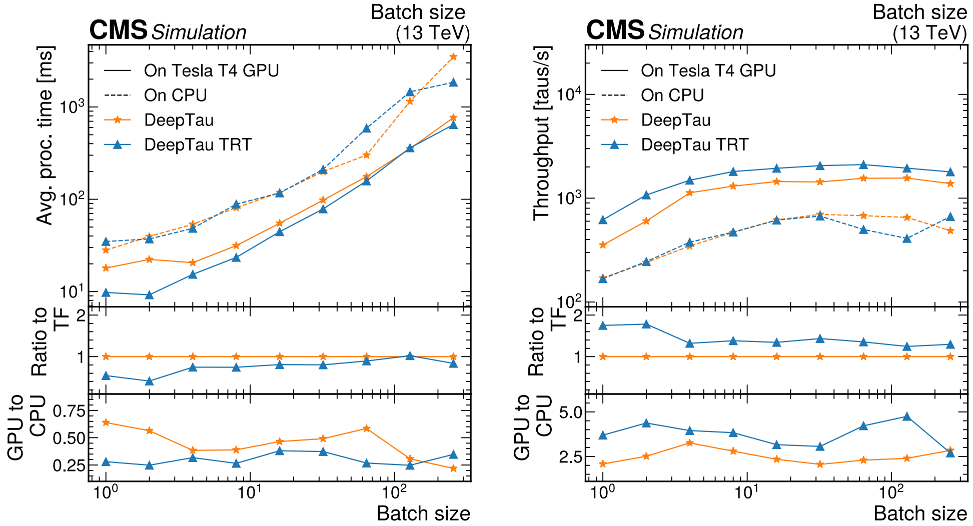


Fig. 7 Average processing time (left) and throughput (right) of the DeepTau algorithm served by a TRITON server running on one NVIDIA Tesla T4 GPU, presented as a function of the batch size. Values are shown for different inference backends: TENSORFLOW (TF) (orange), and TENSORFLOW with TRT (blue). Performance values for these backends when running on a CPU-based TRITON server are given in dashed lines, with the same color-to-backend correspondence



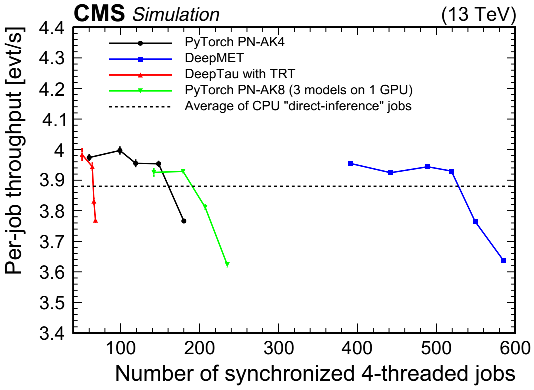


Fig. 8 The GPU saturation scan performed in GCP, where the per-event throughput is shown as a function of the number of parallel CPU clients for the PYTORCH version of PN-AK4 (black), DeepMET (blue), DeepTau optimized with TRT (red), and all PYTORCH versions of PN-AK8 on a single GPU (green). Each of the parallel jobs was run in a four-threaded configuration. The CPU tasks ran in four-threaded GCP VMs, and the TRITON servers were hosted on separate single GPU VMs also in GCP. The line for direct-inference jobs represents the baseline configuration measured by running all algorithms without the use of the SONIC approach or any GPUs. Each solid line represents running one of the specified models on GPU via the SONIC approach

four-threaded client jobs. For DeepTau and DeepMET, a single GPU hosting only one of the algorithms could serve about 64 and 520 client jobs, respectively. The differences between these saturation points are primarily due to different model sizes and numbers of objects per event. From these saturation values, it is possible to determine the number of GPUs needed to serve a production job that uses many client-side CPUs, and thus to determine the ratio of the number of GPUs hosting different models. For ParticleNet and DeepTau, the saturation points will depend on the number of jets and tau leptons in the processed events, such that the ratio of GPUs hosting different models and the required ratio of GPUs to CPUs is dependent on the type of events. If dynamic batching is enabled, the number of GPUs required for a model scales approximately linearly with the number of objects per event that require an ML inference. For example, if there are half as many jets per event, it would require about half as many GPUs to serve ParticleNet, while the number of GPUs required for DeepMET should not change, as that model makes one inference per event.

A single GPU can host multiple models such as DeepTau and DeepMET. In practice, it was found that loading only a single model on each GPU (split-model) led to about a 3–5% increase in the overall performance relative to loading every model on each GPU (all-on-one). If a 10% improvement in throughput is observed in the split-model configuration, one would expect better than a 9.5% improvement in the all-on-one configuration. Because of this slight improvement, the split-model configuration was used in subsequent large-scale tests. However, in scenarios where different physics processes are combined into a single data set, using the all-on-one configuration may be the most straightforward deployment option. Preparatory model profiling should still be performed to ensure that enough GPUs and model instances will be available, but the difference in performance between model partitioning schemes is small.

5.2 Cross-site Tests

A potential bottleneck in the SONIC paradigm is the increased inference latency caused by the physical distance between client and server and other network traffic. While a previous study observed that the average processing time difference between remote and on-premises servers is negligible [63], we tested this observation explicitly with the Mini-AOD workflow, with results presented in Fig. 9. In this test, client-side jobs were executed at Purdue's Tier-2 computing cluster in Indiana. All the models are loaded into one server running on a single GPU for simplicity. The blue points and lines show the throughput improvement in the Mini-AOD workflow when the client jobs communicate with TRITON servers hosting all the models at the same time, running on a single GPU also physically located at Purdue. The improvement is shown as a function of the number of simultaneous four-threaded client-side jobs running at once. The single GPU server begins to saturate when about 10 client-side jobs are sending requests at once and the Mini-AOD workflow running with the SONIC approach becomes slower than the CPU-only workflow if more than about 17 four-threaded client-side jobs are running at once. The direct-inference line in Fig. 9 was made in the same way discussed in "Per-Algorithm Inference Optimization for Fig. 8, though the CPU-only jobs were run at Purdue rather than GCP in Iowa.

These cross-site tests were performed before model configuration optimization as discussed in the preceding section. The exact model configuration is less important for this test, as long as the near and far servers have the same type of GPUs and host the same models. Here, the non-TRT version of DeepTau was used, which saturated close to 40 synchronized four-threaded jobs. When all the models are loaded on a single GPU server, the approximate saturation point can be found with reciprocal addition. With the saturation points of 90 (AK4 jet ParticleNet), 190 (AK8 jet ParticleNet, 3 models), 40 (DeepTau), and 520 (DeepMET, 2 models), the expected saturation point is around 18 synchronized four-threaded jobs, when running all the models together on one GPU. As noted in the previous section, this estimate is often about 5% high. The saturation point seen in Fig. 9 is lower than expected from the above analysis due to the different configurations of the CPU machines in each scenario. Twenty-threaded, hyperthreading-disabled Intel Xeon Processor E5-2660 v3 units were used at Purdue, which are more powerful processors than those used in the GCP-based tests. The faster the client-side resources process events, the lower the saturation point is for a given type of GPU.

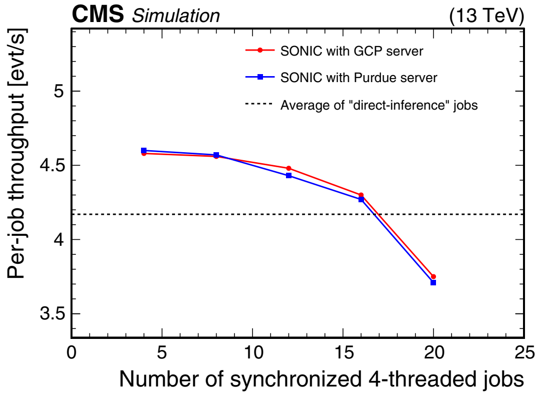


Fig. 9 Production tests across different sites. The CPU tasks always run at Purdue, while the servers with GPU inference tasks for all the models run at Purdue (blue) and at GCP in Iowa (red). The throughput values are higher than those shown in Fig. 8 because the CPUs at Purdue are more powerful than those comprising the GCP VMs

The blue points and line in Fig. 9 show the throughput improvement when the client-side jobs are once again run at Purdue, while the TRITON server hosting all the models on a single GPU is run on GCP resources, which were physically located in Iowa for this test. Both the observed throughput increase and observed GPU saturation point are about the same for both server locations, so we conclude that the client-to-server distance has little impact on performance up to a few hundred kilometers. In the future, it will be important to monitor the impact of distance, especially beyond a few hundred kilometers.

5.3 Large-Scale Tests

Finally, we perform large-scale tests to emulate realistic Mini-AOD production scenarios. These tests were performed exclusively using GCP resources. Here, 24 NVIDIA Tesla T4 GPUs were used to host the PYTORCH version of ParticleNet for AK4 jets, 20 GPUs were used to host the PYTORCH version of all three ParticleNet models for AK8 jets, 48 GPUs were used to host DeepTau, and 10 GPUs were used to host DeepMET. The ratios of the number of GPUs hosting each model do not exactly match those expected from "Per-Algorithm Inference Optimization" section; this was done for multiple reasons. First, a larger number of GPUs than strictly necessary were used in the tests to ensure that the performance would meet the expectations described in "Per-Algorithm Inference Optimization"section, and thereby avoid the cost of repeating the test multiple times. For instance, based on a single-GPU saturation point of 150 four-threaded jobs, one would expect that only about 17 GPUs would be needed to serve the AK4 jet ParticleNet model. It is worth noting that while the demonstration illustrated here uses a CPU-to-GPU ratio of 9820:102 (about 1 GPU per 96 CPU cores), achieving a higher ratio is likely possible as we could have safely decreased the number of GPUs used. Second, it was more practical to instantiate VMs with factors of 4 GPUs. This had the added benefit of maximizing the allowed I/O bandwidth for VMs, which in GCP is restricted for VMs with fewer CPU cores. A maximal CPU-to-GPU ratio for VMs in GCP is achieved for machines with 4 GPUs. While I/O bandwidth was not expected to create problems for this large test, a conservative approach was taken to mitigate the risk of needing repeated trials. Thus, the GPU allocation approach was to find the minimum number of GPUs expected based on single saturation scans (17 for the AK4 jet ParticleNet model, 14 for the AK8 jet ParticleNet models, 42 for DeepTau, and 5 for DeepMET), find the next largest number divisible by 4, then add one extra server corresponding to 4 additional GPUs. For DeepMET, two 4-GPU VMs were used along with one 2-GPU machine, as DeepMET is a relatively lightweight algorithm, and the approach used for the other algorithms would have more than doubled the number of GPUs allocated for the algorithm.

A separate Kubernetes load-balancer was used for each model type to distribute inference requests evenly among server-hosting, GPU-enabled VMs. Thus, client VMs used separate IP addresses for each model type, and each inference for a particular model type was passed through a single load-balancing machine, allowing for network monitoring for each model separately.

Client-side jobs also ran on CPU-only GCP resources, using HEPCloud to dynamically allocate preemptible resources and assign jobs to the client-side VMs. Each client job was run in a four-threaded configuration, with input data files stored locally, and each VM created in this HEPCloud setup had 32 cores and 160 GB of memory, meaning up to 8 simultaneous jobs could run in a single VM.

The largest test had 2500 simultaneous client-side jobs, which amounts to 10 000 CPU cores. Because these jobs were run on preemptible resources, Google reserves the right to reallocate any VM to higher priority requests from other GCP users. Of the 2500 jobs, 2455 jobs completed successfully without preemption, so in total 9820 client-side CPUs were used.

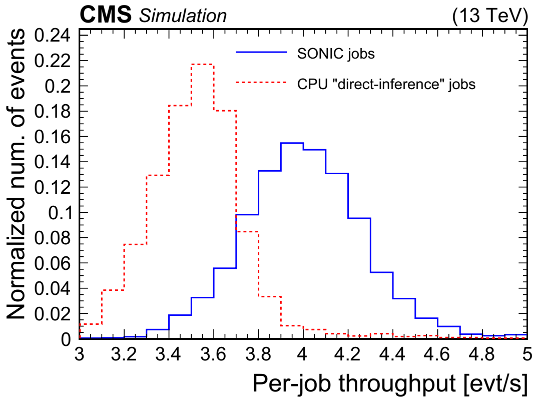


Fig. 10 Scale-out test results on Google Cloud. The average throughput of the workflow with the SONIC approach is 4.0 events/s (solid blue), while the average throughput of the direct-inference workflow is 3.5 events/s (dashed red)

The results of this large-scale test are shown in Fig. 10. The jobs running with the SONIC approach achieved an average throughput of 4.0 events/s, while CPU-only benchmarking jobs had a throughput of 3.5 events/s. This 13% increase in throughput matches the expectation of completely removing the ParticleNet, DeepTau, and DeepMET inference from the total Mini-AOD per-event processing time within the uncertainty, which is typically around 3% from small-scale tests. The throughput values for this test are slightly different from those shown in Figs. 8 and 9. As noted previously, this is due to the difference in the CPUs used in the tests.

As mentioned before, server-side VMs were optimized to allow maximal input and output bandwidth. No bottlenecks due to bandwidth were observed in this scale-out test. We noted that the maximum data input rate received by one of the Kubernetes load balancers was 11.5 GB/s, which was for DeepTau. The next-highest data input rate was 3.3 GB/s for DeepMET, and less than 500 MB/s of input was needed for all ParticleNet models combined. These values are consistent with expectations from Table 2, as there were roughly 10 000 CPU cores running simultaneously, each processing about 1 event per second. The output rate was significantly smaller for each model, as most return only one or a few floating-point values as the inference result. In the future, more algorithms will be able to be run on coprocessors. These could have larger I/O sizes than the algorithms considered here, so continuing to monitor the network usage will be important.

6 Portability

While the inference servers discussed so far have exclusively utilized GPU resources, servers are easily portable and can run on other processing platforms. Previous uses of the SONIC approach with FPGAs are reported in detail in Refs.

[61,62], where the portability was demonstrated by changing the coprocessor technology and server backend without modifying the client-side software. In the new studies reported here, we have additionally run the Mini-AOD workflow with servers using both CPUs and Graphcore IPUs [33,34].

6.1 The CPU Fallback Server

When running with remote servers, one potentially common and important failure mode is communication errors between clients and servers. To support automatic local CPU inference as a backup option when communication failures occur with remote servers, the SONIC implementation includes a service that can launch a Triton server using local CPU resources for any SONIC approach-compatible models, referred to as a fallback server. Fallback servers can also be used for inference when third-party ML frameworks are not supported for direct inference in cmssw.

Ideally, the use of fallback servers should have minimal impact on per-event throughput relative to running direct-inference jobs without the SONIC approach. This is contingent on two factors. First, the latency introduced by sending data to or from local servers must be negligible. Second, servers should introduce minimal overhead to maximize the CPU resources used to perform inference. The first concern can be resolved using the shared-memory option, which skips the gRPC communication and directly passes the data in certain memory chunks between the server and client. The gRPC overhead in most cases is found to be negligible. Regarding the second concern, local servers are running on the same CPUs as the other modules in the workflow, so scheduling efforts must be made to avoid CPU thread contention between the two. This implies that the synchronous server mode is preferred for local CPU fallback servers. Additionally, inference tasks should not create extra threads to avoid contention. In our experiments, we found that using more inference threads in the server than the number of threads allocated for the cmssw job will slow processing down dramatically.

Having thus explored potential configurations, for the local CPU inference, we run our servers in synchronous mode, with the number of model instances set to the number of threads per job, and the number of inference threads always set to one. This configuration mimics direct inference and avoids thread over-subscription as much as possible. We compare the throughput between direct inference and the SONIC approach with local CPU fallback servers using this configuration. Tests were performed using resources at the Purdue Tier-2 cluster with the CPU-only nodes. There are $n_{\mathrm{CPU}} = 20$ Intel E5-2660 CPU cores on one node, and hyperthreading is disabled to ensure more stable results. For all tests, we always saturate the CPU nodes by requiring the product of the number of jobs (n_{j}) and number of threads per job (n_{T}) to be equal to the number of CPU cores: $n_{\mathrm{j}} n_{\mathrm{T}} = n_{\mathrm{CPU}} = 20$. For example, to test a configuration where a single job occupies four threads ($n_{\mathrm{T}} = 4$), we would run five synchronized jobs ($n_{\mathrm{j}} = 5$), while a two-threaded configuration would require 10 synchronized jobs. We scan the throughput as a function of the number of threads, as shown in Fig. 11. The throughput of running with local CPU fallback servers is similar to direct inference. The higher throughput in some cases is a result of optimizations in more recent versions of ONNX Runtime installed on the server, which can be controlled in real production jobs.

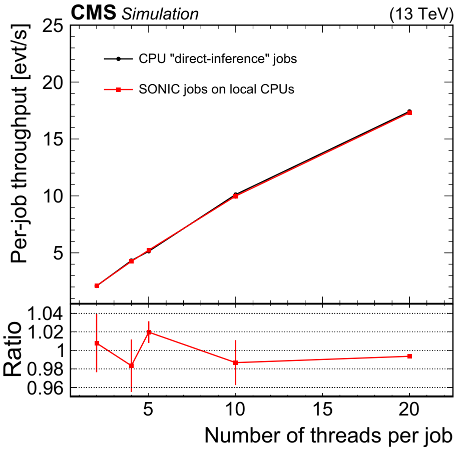


Fig. 11 Throughput (upper) and throughput ratio (lower) between the SONIC approach and direct inference in the local CPU tests at the Purdue Tier-2 cluster. To ensure the CPUs are always saturated, the number of threads per job multiplied by the number of jobs is set to 20

Memory usage was also monitored with the top command during the studies and provided in Table 3. As the number of threads per job increases, the number of model instances increases as well, leading to higher memory usage in the SONIC approach compared with direct inference. On the other hand, if the number of model instances is fixed to one, the server memory usage is always around 300 MB, so the total memory usage is similar to direct inference, but the throughput decreases by 5–10%, depending on the tasks. Further simultaneous optimization of the throughput and memory usage will be explored in the future.

6.2 Studies with Graphcore IPUs

As discussed in Sects. 3.2 and 3.3, NVIDIA Triton inference servers support custom backends to run with different coprocessors and different (e.g.,, ML) backends. Since the SONIC approach's client code only depends on the Triton protocols, algorithms implemented in this way can easily be ported to different types of coprocessors. One of the Mini-AOD production tests was run together with the Graphcore IPU team, where a custom backend was prepared by the developer team to support running ML inference with IPUs.

Table 3 Memory usage with direct inference and the SONIC approach in the local CPU tests at the Purdue Tier-2 cluster. The last column is calculated as the sum of client and server memory usage divided by the direct inference memory usage. To ensure the CPUs are always saturated, the number of threads n_T per job multiplied by the number of jobs is set to 20

n_T per job	CMSSW with direct inference [MB]	CMSSW with SONIC app. (client) [MB]	SONIC app server [MB]	SONIC/direct
1	1850	1700	200	103%
2	2000	1800	400	110%
5	2200	1950	800	125%
10	2500	2200	1200	136%
20	2900	2500	2000	155%

The supported ML frameworks on IPUs when the tests were performed included TensorFlow, ONNX, and PyTorch. PyTorch- Geometric support was in development at that time and is now available. With TensorFlow custom backend available for the server, we can easily run DeepMET and DeepTau inference with the SONIC approach in CMSSW on IPUs.

First, a per-algorithm throughput scan was carried out. Comparing running inference on MK2 GC200 IPUs with NVIDIA Tesla V100s, a chip-to-chip factor of 3 throughput improvement was found, with larger gains expected for more computing-intensive models. Running the entire Mini-AOD workflow was also tested. We adapted the workflow configuration to point the CMSSW clients running at the Purdue Tier-2 cluster to IPU servers on the Graphcloud cloud. Without any other modifications on the client side, the workflow ran successfully, performing inference of the DeepMET and DeepTau models on the cloud and the other parts of the workflow on the client local CPUs. Outputs were checked and found to be consistent with direct inference, within 10^{-6} differences for floats due to precision limits.

7 Summary

Within the next decade, the data-taking rate at the LHC will increase dramatically, straining the expected computing resources of the LHC experiments. At the same time, more algorithms that run on these resources will be converted into either machine learning or domain algorithms that are easily accelerated with the use of coprocessors, such as graphics processing units (GPUs). By pursuing heterogeneous architectures, it is possible to alleviate potential shortcomings of available central processing unit (CPU) resources.

Inference as a service (IaaS) is a promising scheme to integrate coprocessors into CMS computing workflows. In IaaS, client code simply assembles the input data for an algorithm, sends that input to an inference server running either locally or remotely, and retrieves output from the server. The implementation of IaaS discussed throughout this paper is called the Services for Optimized Network Inference on Coprocessors (SONIC) approach, which employs NVIDIA Triton Inference Servers to host models on coprocessors, as demonstrated here in studies on GPUs, CPUs, and Graphcore Intelligence Processing Units (IPUs).

In this paper, the SONIC approach in the CMS software framework (CMSSW) is demonstrated in a sample Mini-AOD workflow, where algorithms for jet tagging, tau lepton identification, and missing transverse momentum regression are ported to run on inference servers. These algorithms account for nearly 10% of the total processing time per event in a simulated data set of top quark-antiquark events. After model profiling, which is used to optimize server performance and determine the needed number of GPUs for a given number of client jobs, the expected 10% decrease in per-event processing time was achieved in a large-scale test of Mini-AOD production with the SONIC approach that used about 10 000 CPU cores and 100 GPUs. The network bandwidth is large enough to support high input–output model inference for the workflow tested, and it will be monitored as the fraction of algorithms using remote GPUs increases.

In addition to meeting performance expectations, we demonstrated that the throughput results are not highly sensitive to the physical client-to-server distance, at least up to distances of hundreds of kilometers. Running inference through Triton servers on local CPU resources does not affect the throughput compared with the standard approach of running inference directly on CPUs in the job thread. We also performed a test using GraphCore IPUs to demonstrate the flexibility of the SONIC approach.

The SONIC approach for IaaS represents a flexible method to accelerate algorithms, which is increasingly valuable for LHC experiments. Using a realistic workflow, we highlighted many of the benefits of the SONIC approach, including the use of remote resources, workflow acceleration, and portability to different processor technologies. To make it a viable and robust paradigm for CMS computing in the future, additional studies are ongoing or planned for monitoring and mitigating potential issues such as excessive network and memory usage or server failures.

Acknowledgements We congratulate our colleagues in the CERN accelerator departments for the excellent performance of the LHC and

thank the technical and administrative staffs at CERN and at other CMS institutes for their contributions to the success of the CMS effort. In addition, we gratefully acknowledge the computing centers and personnel of the Worldwide LHC Computing Grid and other centers for delivering so effectively the computing infrastructure essential to our analyses. Finally, we acknowledge the enduring support for the construction and operation of the LHC, the CMS detector, and the supporting computing infrastructure provided by the following funding agencies: SC (Armenia), BMBWF and FWF (Austria); FNRS and FWO (Belgium); CNPq, CAPES, FAPERJ, FAPERGS, and FAPESP (Brazil); MES and BNSF (Bulgaria); CERN; CAS, MoST, and NSFC (China); MINCIENCIAS (Colombia); MSES and CSF (Croatia); RIF (Cyprus); SENESCYT (Ecuador); ERC PRG, RVTT3 and TK202 (Estonia); Academy of Finland, MEC, and HIP (Finland); CEA and CNRS/IN2P3 (France); SRNSF (Georgia); BMBF, DFG, and HGF (Germany); GSRI (Greece); NKFIH (Hungary); DAE and DST (India); IPM (Iran); SFI (Ireland); INFN (Italy); MSIP and NRF (Republic of Korea); MES (Latvia); LAS (Lithuania); MOE and UM (Malaysia); BUAP, CINVESTAV, CONACYT, LNS, SEP, and UASLP-FAI (Mexico); MOS (Montenegro); MBIE (New Zealand); PAEC (Pakistan); MES and NSC (Poland); FCT (Portugal); MESTD (Serbia); MCIN/AEI and PCTI (Spain); MOSTR (Sri Lanka); Swiss Funding Agencies (Switzerland); MST (Taipei); MHESI and NSTDA (Thailand); TUBITAK and TENMAK (Turkey); NASU (Ukraine); STFC (United Kingdom); DOE and NSF (USA).
Individuals have received support from the Marie-Curie programme and the European Research Council and Horizon 2020 Grant, contract Nos. 675440, 724704, 752730, 758316, 765710, 824093, and COST Action CA16108 (European Union); the Leventis Foundation; the Alfred P. Sloan Foundation; the Alexander von Humboldt Foundation; the Science Committee, project no. 22rl-037 (Armenia); the Belgian Federal Science Policy Office; the Fonds pour la Formation à la Recherche dans l'Industrie et dans l'Agriculture (FRIA-Belgium); the Agentschap voor Innovatie door Wetenschap en Technologie (IWT-Belgium); the F.R.S.-FNRS and FWO (Belgium) under the "Excellence of Science—EOS" – be.h project n. 30820817; the Beijing Municipal Science & Technology Commission, No. Z191100007219010 and Fundamental Research Funds for the Central Universities (China); the Ministry of Education, Youth and Sports (MEYS) of the Czech Republic; the Shota Rustaveli National Science Foundation, grant FR-22-985 (Georgia); the Deutsche Forschungsgemeinschaft (DFG), under Germany's Excellence Strategy – EXC 2121 "Quantum Universe"—390833306, and under project number 400140256—GRK2497; the Hellenic Foundation for Research and Innovation (HFRI), Project Number 2288 (Greece); the Hungarian Academy of Sciences, the New National Excellence Program - ÚNKP, the NKFIH research grants K 124845, K 124850, K 128713, K 128786, K 129058, K 131991, K 133046, K 138136, K 143460, K 143477, 2020−2.2.1-ED-2021-00181, and TKP2021-NKTA-64 (Hungary); the Council of Science and Industrial Research, India; ICSC—National Research Centre for High Performance Computing, Big Data and Quantum Computing, funded by the EU NexGeneration program (Italy); the Latvian Council of Science; the Ministry of Education and Science, project no. 2022/WK/14, and the National Science Center, contracts Opus 2021/41/B/ST2/01369 and 2021/43/B/ST2/01552 (Poland); the Fundação para a Ciência e a Tecnologia, grant CEECIND/01334/2018 (Portugal); the National Priorities Research Program by Qatar National Research Fund; MCIN/AEI/10.13039/501100011033, ERDF "a way of making Europe", and the Programa Estatal de Fomento de la Investigación Científica y Técnica de Excelencia María de Maeztu, grant MDM-2017-0765 and Programa Severo Ochoa del Principado de Asturias (Spain); the Chulalongkorn Academic into Its 2nd Century Project Advancement Project, and the National Science, Research and Innovation Fund via the Program Management Unit for Human Resources & Institutional Development, Research and Innovation, grant B37G660013 (Thailand); the Kavli Foundation; the Nvidia Corporation; the SuperMicro Corporation; the Welch Foundation, contract C-1845; and the Weston Havens Foundation (USA).

Author contributions N/A: the collaboration submits as a whole.

Funding Open access funding provided by CERN (European Organization for Nuclear Research)

Data Availability No datasets were generated or analyzed during the current study.

Declarations

Conflict of interest The authors declare no conflict of interest.

References

1. Evans L, Bryant P (2008) LHC machine. JINST 3:S08001. https://doi.org/10.1088/1748-0221/3/08/S08001
2. ATLAS Collaboration (2008) The ATLAS experiment at the CERN Large Hadron Collider. JINST 3:S08003. https://doi.org/10.1088/1748-0221/3/08/S08003
3. CMS Collaboration (2008) The CMS experiment at the CERN LHC. JINST 3:S08004. https://doi.org/10.1088/1748-0221/3/08/S08004
4. ATLAS Collaboration (2012) Observation of a new particle in the search for the Standard Model Higgs boson with the ATLAS detector at the LHC. Phys Lett B 716:1. https://doi.org/10.1016/j.physletb.2012.08.020
5. CMS Collaboration (2012) Observation of a new boson at a mass of 125 GeV with the CMS experiment at the LHC. Phys Lett B 716:30. https://doi.org/10.1016/j.physletb.2012.08.021
6. Collaboration CMS, CMS Collaboration (2013) Observation of a new boson with mass near 125 GeV in pp collisions at $\sqrt{s} = 7$ and 8 TeV. JHEP 06:081. https://doi.org/10.1007/JHEP06(2013)081
7. Collaboration CMS, CMS Collaboration (2019) Search for supersymmetry in proton-proton collisions at 13 TeV in final states with jets and missing transverse momentum. JHEP 10:244. https://doi.org/10.1007/JHEP10(2019)244
8. CMS Collaboration (2021) Combined searches for the production of supersymmetric top quark partners in proton-proton collisions at $\sqrt{s} = 13$ TeV. Eur Phys J C 81:970. https://doi.org/10.1140/epjc/s10052-021-09721-5
9. Collaboration CMS (2022) Search for higgsinos decaying to two Higgs bosons and missing transverse momentum in proton-proton collisions at $\sqrt{s} = 13$ TeV. JHEP 05:014. https://doi.org/10.1007/JHEP05(2022)014
10. CMS Collaboration (2021) Search for supersymmetry in final states with two oppositely charged same-flavor leptons

and missing transverse momentum in proton-proton collisions at $\sqrt{s} = 13$ TeV. JHEP 04:123. https://doi.org/10.1007/JHEP04(2021)123

11. ATLAS Collaboration (2019) Search for squarks and gluinos in final states with hadronically decaying τ-leptons, jets, and missing transverse momentum using pp collisions at $\sqrt{s} = 13$ TeV with the ATLAS detector. Phys Rev D 99:012009. https://doi.org/10.1103/PhysRevD.99.012009
12. Collaboration ATLAS (2020) Search for top squarks in events with a Higgs or Z boson using 139 fb^{-1} of pp collision data at $\sqrt{s} = 13$ TeV with the ATLAS detector. Eur Phys J C 80:1080. https://doi.org/10.1140/epjc/s10052-020-08469-8
13. ATLAS Collaboration (2021) Search for charginos and neutralinos in final states with two boosted hadronically decaying bosons and missing transverse momentum in pp collisions at $\sqrt{s} = 13$ TeV with the ATLAS detector. Phys Rev D 104:112010. https://doi.org/10.1103/PhysRevD.104.112010
14. ATLAS Collaboration (2023) Search for direct pair production of sleptons and charginos decaying to two leptons and neutralinos with mass splittings near the W-boson mass in $\sqrt{s} = 13$ TeV pp collisions with the ATLAS detector. JHEP 06:031. https://doi.org/10.1007/JHEP06(2023)031
15. ATLAS Collaboration (2021) Search for new phenomena in events with an energetic jet and missing transverse momentum in pp collisions at $\sqrt{s} = 13$ TeV with the ATLAS detector. Phys Rev D 103:112006. https://doi.org/10.1103/PhysRevD.103.112006
16. CMS Collaboration (2021) Search for new particles in events with energetic jets and large missing transverse momentum in proton-proton collisions at $\sqrt{s} = 13$ TeV. JHEP 11:153. https://doi.org/10.1007/jhep11(2021)153
17. ATLAS Collaboration (2020) Search for new resonances in mass distributions of jet pairs using 139 fb^{-1} of pp collisions at $\sqrt{s} = 13$ TeV with the ATLAS detector. JHEP 03:145. https://doi.org/10.1007/jhep03(2020)145
18. ATLAS Collaboration (2019) Search for high-mass dilepton resonances using 139 fb^{-1} of pp collision data collected at $\sqrt{s} = 13$ TeV with the ATLAS detector. Phys Lett B 796:68. https://doi.org/10.1016/j.physletb.2019.07.016
19. Collaboration CMS (2018) Search for narrow and broad DIJET resonances in proton-proton collisions at $\sqrt{s} = 13$ TeV and constraints on dark matter mediators and other new particles. JHEP 08:130. https://doi.org/10.1007/jhep08(2018)130
20. Collaboration CMS (2020) Search for high mass dijet resonances with a new background prediction method in proton-proton collisions at $\sqrt{s} = 13$ TeV. JHEP 05:033. https://doi.org/10.1007/jhep05(2020)033
21. CMS Collaboration (2021) Search for resonant and nonresonant new phenomena in high-mass dilepton final states at $\sqrt{s} = 13$ TeV. JHEP 07:208. https://doi.org/10.1007/jhep07(2021)208
22. Aberle O et al (2020) High-Luminosity Large Hadron Collider (HL-LHC): Technical design report, CERN Yellow Rep Monogr https://doi.org/10.23731/CYRM-2020-0010
23. Bruning O, Rossi L (2015) The High Luminosity Large Hadron Collider: the new machine for illuminating the mysteries of universe. World Sci. https://doi.org/10.1142/9581
24. CMS Collaboration (2020) The Phase-2 upgrade of the CMS level-1 trigger, CMS Technical Design Report CERN-LHCC-2020-004, CMS-TDR-021. https://cds.cern.ch/record/2714892
25. ATLAS Collaboration (2017) Technical design report for the Phase-II upgrade of the ATLAS TDAQ system, ATLAS Technical Design Report CERN-LHCC-2017-020, ATLAS-TDR-029. https://doi.org/10.17181/CERN.2LBB.4IAL
26. Ryd A, Skinnari L (2020) Tracking triggers for the HL-LHC. Ann Rev Nucl Part Sci 70:171. https://doi.org/10.1146/annurev-nucl-020420-093547. arXiv:2010.13557
27. Collaboration A, ATLAS Collaboration (2020) Operation of the ATLAS trigger system in Run 2. JINST 15:P10004. https://doi.org/10.1088/1748-0221/15/10/P10004
28. CMS Collaboration (2017) The CMS trigger system. JINST 12:01020. https://doi.org/10.1088/1748-0221/12/01/P01020
29. CMS Collaboration (2021) The Phase-2 upgrade of the CMS data acquisition and high level trigger, CMS Technical Design Report CERN-LHCC-2021-007, CMS-TDR-022, 2021. https://cds.cern.ch/record/2759072
30. CMS Offline Software and Computing Group (2022) CMS Phase-2 computing model: Update document, CMS Note CMS-NOTE-2022-008.https://cds.cern.ch/record/2815292
31. ATLAS Collaboration (2022) ATLAS software and computing HL-LHC roadmap, LHCC Public Document CERN-LHCC-2022-005, LHCC-G-182. http://cds.cern.ch/record/2802918
32. Dennard RH et al (1974) Design of ion-implanted MOSFET's with very small physical dimensions. IEEE J Solid-State Circuits 9:256. https://doi.org/10.1109/JSSC.1974.1050511
33. Graphcore Intelligence processing unit. https://www.graphcore.ai/products/ipu. Accessed 08 Nov 2023
34. Jia Z, Tillman B, Maggioni M, Scarpazza DP (2019) Dissecting the Graphcore IPU architecture via microbenchmarking, arXiv:1912.03413
35. Guest D, Cranmer K, Whiteson D (2018) Deep learning and its application to LHC physics. Ann Rev Nucl Part Sci 68:161. https://doi.org/10.1146/annurev-nucl-101917-021019. arXiv:1806.11484
36. Albertsson K et al (2018) Machine learning in high energy physics community white paper. J Phys Conf Ser 1085:022008. https://doi.org/10.1088/1742-6596/1085/2/022008. arXiv:1807.02876
37. Bourilkov D (2020) Machine and deep learning applications in particle physics. Int J Mod Phys A 34:1930019. https://doi.org/10.1142/S0217751X19300199. arXiv:1912.08245
38. Larkoski AJ, Moult I, Nachman B (2020) Jet substructure at the Large Hadron Collider: a review of recent advances in theory and machine learning. Phys Rept 841:1. https://doi.org/10.1016/j.physrep.2019.11.001. arXiv:1709.04464
39. Matthew F, Benjamin N (2021) A living review of machine learning for particle physics, arXiv:2102.02770
40. Harris P, et al (2022) Physics community needs, tools, and resources for machine learning. In: Proceedings 2021 US Community Study on the Future of Particle Physics. arXiv:2203.16255
41. Savard C, et al (2023) Optimizing high throughput inference on graph neural networks at shared computing facilities with the NVIDIA Triton Inference Server, 12. arXiv:2312.06838
42. Farrell S, et al (2018) Novel deep learning methods for track reconstruction, In: 4th International Workshop Connecting The Dots 2018. arXiv:1810.06111
43. Amrouche S, et al (2019) The Tracking Machine Learning challenge: accuracy phase, ch 9, p 231. Springer Cham, 4, https://doi.org/10.1007/978-3-030-29135-8_9
44. Ju X et al (2021) Performance of a geometric deep learning pipeline for HL-LHC particle tracking. Eur Phys J C 81:876. https://doi.org/10.1140/epjc/s10052-021-09675-8. arXiv:2103.06995
45. DeZoort G et al (2021) Charged particle tracking via edge-classifying interaction networks. Comput Softw Big Sci 5:26. https://doi.org/10.1007/s41781-021-00073-z. arXiv:2103.16701
46. Qasim SR, Kieseler J, Iiyama Y, Pierini M (2019) Learning representations of irregular particle-detector geometry with distance-weighted graph networks. Eur Phys J C 79:608. https://doi.org/10.1140/epjc/s10052-019-7113-9. arXiv:1902.07987
47. Kieseler J (2020) Object condensation: one-stage grid-free multi-object reconstruction in physics detectors, graph and image data. Eur Phys J C 80:886. https://doi.org/10.1140/epjc/s10052-020-08461-2. arXiv:2002.03605

48. CMS Collaboration (2023) GNN-based end-to-end reconstruction in the CMS Phase 2 high-granularity calorimeter. J Phys Conf Ser. 2438:012090. https://doi.org/10.1088/1742-6596/2438/1/012090
49. Pata J et al (2021) MLPF: Efficient machine-learned particle-flow reconstruction using graph neural networks. Eur Phys J C 81:381. https://doi.org/10.1140/epjc/s10052-021-09158-w. arXiv:2101.08578
50. CMS Collaboration (2023) Machine learning for particle flow reconstruction at CMS. J Phys Conf Ser 2438:012100. https://doi.org/10.1088/1742-6596/2438/1/012100
51. Mokhtar F, et al (2023) Progress towards an improved particle flow algorithm at CMS with machine learning. In: Proceedings 21st International Workshop on Advanced Computing and Analysis Techniques in Physics Research: AI meets Reality
52. Di Bello FA et al (2023) Reconstructing particles in jets using set transformer and hypergraph prediction networks. Eur Phys J C 83:596. https://doi.org/10.1140/epjc/s10052-023-11677-7. arXiv:2212.01328
53. Pata J et al (2023) Improved particle-flow event reconstruction with scalable neural networks for current and future particle detectors. arXiv:2309.06782
54. Moreno EA et al (2020) JEDI-net: a jet identification algorithm based on interaction networks. Eur Phys J C 80:58. https://doi.org/10.1140/epjc/s10052-020-7608-4. arXiv:1908.05318
55. Qu H, Gouskos L (2020) ParticleNet: jet tagging via particle clouds. Phys Rev D 101:056019. https://doi.org/10.1103/PhysRevD.101.056019. arXiv:1902.08570
56. Moreno EA et al (2020) Interaction networks for the identification of boosted $\mathrm{H} \rightarrow b\bar{b}$ decays. Phys Rev D 102:012010. https://doi.org/10.1103/PhysRevD.102.012010. arXiv:1909.12285
57. Bols E et al (2020) Jet flavour classification using DeepJet. JINST 15:P12012. https://doi.org/10.1088/1748-0221/15/12/P12012. arXiv:2008.10519
58. CMS Collaboration (2020) Identification of heavy, energetic, hadronically decaying particles using machine-learning techniques. JINST 15:P06005. https://doi.org/10.1088/1748-0221/15/06/P06005
59. Qu H, Li C, Qian S (2022) Particle transformer for jet tagging. In: Proceedings 39th International Conference on Machine Learning, Chaudhuri K et al. eds. vol. 162, p 18281
60. CMS Collaboration (2023) Muon identification using multivariate techniques in the CMS experiment in proton-proton collisions at $\sqrt{s} = 13\,\mathrm{TeV}$. arXiv:2310.03844
61. Duarte J et al (2019) FPGA-accelerated machine learning inference as a service for particle physics computing. Comput Softw Big Sci 3:13. https://doi.org/10.1007/s41781-019-0027-2. arXiv:1904.08986
62. Rankin D et al (2020) FPGAs-as-a-service toolkit (FaaST), In: Proceedings 2020 IEEE/ACM International Workshop on Heterogeneous High-performance Reconfigurable Computing (H2RC). IEEE, https://doi.org/10.1109/h2rc51942.2020.00010
63. Krupa J et al (2021) GPU coprocessors as a service for deep learning inference in high energy physics. Mach Learn Sci Tech 2:035005. https://doi.org/10.1088/2632-2153/abec21. arXiv:2007.10359
64. Wang M et al (2021) GPU-accelerated machine learning inference as a service for computing in neutrino experiments. Front Big Data 3:604083. https://doi.org/10.3389/fdata.2020.604083. arXiv:2009.04509
65. Cai T et al (2023) Accelerating machine learning inference with GPUs in ProtoDUNE data processing. Comput Softw Big Sci 7:11. https://doi.org/10.1007/s41781-023-00101-0. arXiv:2301.04633
66. Gunny A et al (2022) Hardware-accelerated inference for real-time gravitational-wave astronomy. Nature Astron 6:529. https://doi.org/10.1038/s41550-022-01651-w. arXiv:2108.12430
67. ALICE Collaboration (2019) Real-time data processing in the ALICE high level trigger at the LHC. Comput Phys Commun 242:25. https://doi.org/10.1016/j.cpc.2019.04.011
68. Aaij R et al (2020) Allen: a high level trigger on GPUs for LHCb. Comput Softw Big Sci 4:7. https://doi.org/10.1007/s41781-020-00039-7. arXiv:1912.09161
69. LHCb Collaboration (2023) The LHCb upgrade I. arXiv:2305.10515
70. Bocci A et al (2020) Heterogeneous reconstruction of tracks and primary vertices with the CMS pixel tracker. Front Big Data 3:601728. https://doi.org/10.3389/fdata.2020.601728. arXiv:2008.13461
71. Collaboration CMS (2023) CMS high level trigger performance comparison on CPUs and GPUs. J Phys Conf Ser 2438:012016. https://doi.org/10.1088/1742-6596/2438/1/012016
72. Vom Bruch D (2020) Real-time data processing with GPUs in high energy physics. JINST 15:C06010. https://doi.org/10.1088/1748-0221/15/06/C06010. arXiv:2003.11491
73. Collaboration CMS (2015) Mini-AOD: A new analysis data format for CMS. J Phys Conf Ser 664:7. https://doi.org/10.1088/1742-6596/664/7/072052
74. CMS Collaboration (2006) CMS physics: Technical design report volume 1: Detector performance and software. CMS Technical Design Report CERN-LHCC-2006-001, CMS-TDR-8-1. https://cds.cern.ch/record/922757
75. CMS Collaboration CMSSW on Github. http://cms-sw.github.io/ Accessed 08 Nov 2023
76. CMS Collaboration (2020) Performance of the CMS Level-1 trigger in proton-proton collisions at $\sqrt{s} = 13\,\mathrm{TeV}$. JINST 15:P10017. https://doi.org/10.1088/1748-0221/15/10/P10017
77. CMS Collaboration (2021) Electron and photon reconstruction and identification with the CMS experiment at the CERN LHC. JINST 16:P05014. https://doi.org/10.1088/1748-0221/16/05/P05014
78. Collaboration CMS, CMS Collaboration (2014) Description and performance of track and primary-vertex reconstruction with the CMS tracker. JINST 9:P10009. https://doi.org/10.1088/1748-0221/9/10/P10009
79. CMS Collaboration (2017) Particle-flow reconstruction and global event description with the CMS detector. JINST 12:P10003. https://doi.org/10.1088/1748-0221/12/10/P10003
80. oneTBB, oneAPI Threading Building Blocks. https://github.com/oneapi-src/oneTBB. Accessed 08 Nov 2023
81. Bocci A et al (2020) Bringing heterogeneity to the CMS software framework. Eur Phys J Web Conf 245:05009. https://doi.org/10.1051/epjconf/202024505009. arXiv:2004.04334
82. CMS Collaboration (2019) A further reduction in CMS event data for analysis: the NANOAOD format. Eur Phys J Web Conf 214:06021. https://doi.org/10.1051/epjconf/201921406021
83. Collaboration CMS (2020) Extraction and validation of a new set of CMS pythia 8 tunes from underlying-event measurements. Eur Phys J C 80:4. https://doi.org/10.1140/epjc/s10052-019-7499-4
84. Collaboration CMS, CMS Collaboration (2020) Pileup mitigation at CMS in 13 TeV data. JINST 15:P09018. https://doi.org/10.1088/1748-0221/15/09/P09018
85. Collaboration CMS (2020) NANOAOD: a new compact event data format in CMS. Eur Phys J Web Conf. 245:06002. https://doi.org/10.1051/epjconf/202024506002
86. NVIDIA. NVIDIA Triton Inference Server. https://docs.nvidia.com/deeplearning/triton-inference-server/user-guide/docs/index.html. Accessed 08 Nov 2023
87. gRPC, gRPC – a high performance, open source universal RPC framework. https://grpc.io/. Accessed 08 Nov 2023

88. Kubernetes Kubernetes documentation. https://kubernetes.io/docs/home/. Accessed 08 Nov 2023
89. K. Pedro et al SonicCore. https://github.com/cms-sw/cmssw/tree/master/HeterogeneousCore/SonicCore. Accessed 08 Nov 2023
90. K. Pedro et al. SonicTriton. https://github.com/cms-sw/cmssw/tree/master/HeterogeneousCore/SonicTriton. Accessed 08 Nov 2023
91. Pedro K SonicCMS. https://github.com/fastmachinelearning/SonicCMS. Accessed 08 Nov 2023
92. Caulfield AM, et al (2016) A cloud-scale acceleration architecture. In: Proc.eedings 2016 49th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO), p 1. https://doi.org/10.1109/MICRO.2016.7783710
93. Kuznetsov V (2018) vkuznet/TFaaS: First public version, https://doi.org/10.5281/zenodo.1308049
94. Kuznetsov V, Giommi L, Bonacorsi D (2021) MLaaS4HEP: machine learning as a service for HEP. Comput Softw Big Sci 5:17. https://doi.org/10.1007/s41781-021-00061-3. arXiv:2007.14781
95. KServe, KServe documentation website. https://kserve.github.io/website/. Accessed 08 Nov 2023
96. NVIDIA NVIDIA Triton Inference Server README (release 22.08). https://github.com/triton-inference-server/server/blob/r22.08/README.md#documentation Accessed 08 Nov 2023
97. Paszke A et al (2019) PyTorch: an imperative style, high-performance deep learning library, In: Advances in Neural Information Processing Systems 32, Wallach H et al., eds., p 8024. Curran Associates, Inc., http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf. arXiv:1912.01703
98. NVIDIA NVIDIA TensorRT. https://developer.nvidia.com/tensorrt. Accessed 08 Nov 2023
99. ONNX Open Neural Network Exchange (ONNX). https://github.com/onnx/onnx. Accessed 08 Nov 2023
100. Abadi M et al (2016) TensorFlow: A system for large-scale machine learning, arXiv:1605.08695
101. Chen T, Guestrin C (2016) XGBoost: A scalable tree boosting system, In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Association for Computing Machinery. https://doi.org/10.1145/2939672.2939785
102. NVIDIA NVIDIA Triton Inference Server Model Analyzer. https://github.com/triton-inference-server/model_analyzer. Accessed 08 Nov 2023
103. Buncic P et al (2010) CernVM - a virtual software appliance for LHC applications. J Phys Conf Ser 219:042003. https://doi.org/10.1088/1742-6596/219/4/042003
104. Guida SD et al (2015) The CMS condition database system. J Phys Conf Ser 664:042024. https://doi.org/10.1088/1742-6596/664/4/042024
105. Bauerdick L et al (2012) Using Xrootd to federate regional storage. J Phys Conf Ser 396:042009. https://doi.org/10.1088/1742-6596/396/4/042009
106. CMS Collaboration (2019) Performance of missing transverse momentum reconstruction in proton-proton collisions at $\sqrt{s} = 13\,\mathrm{TeV}$ using the CMS detector. JINST 14:P07004. https://doi.org/10.1088/1748-0221/14/07/P07004
107. CMSSW, ParticleNet producer in CMSSW. https://github.com/cms-sw/cmssw/blob/CMSSW_13_0_0/RecoBTag/ONNXRuntime/plugins/BoostedJetONNXJetTagsProducer.cc. Accessed: 08 Nov 2023
108. CMSSW ParticleNet SONIC producer in CMSSW. https://github.com/cms-sw/cmssw/blob/CMSSW_13_0_0/RecoBTag/ONNXRuntime/plugins/ParticleNetSonicJetTagsProducer.cc. Accessed 08 Nov 2023
109. Cacciari M, Salam GP, Soyez G (2008) The anti-k_{T} jet clustering algorithm. JHEP 04:063. https://doi.org/10.1088/1126-6708/2008/04/063. arXiv:0802.1189
110. Cacciari M, Salam GP, Soyez G (2012) FastJet user manual. Eur Phys J C 72:1896. https://doi.org/10.1140/epjc/s10052-012-1896-2. arXiv:1111.6097
111. CMS Collaboration (2023) Performance of the ParticleNet tagger on small and large-radius jets at high level trigger in Run 3. CMS Detector Performance Note CMS-DP-2023-021, https://cds.cern.ch/record/2857440
112. CMS Collaboration (2020) Identification of highly Lorentz-boosted heavy particles using graph neural networks and new mass decorrelation techniques, CMS Detector Performance Note CMS-DP-2020-002, https://cds.cern.ch/record/2707946
113. CMS Collaboration (2021) Mass regression of highly-boosted jets using graph neural networks, CMS Detector Performance Note CMS-DP-2021-017, https://cds.cern.ch/record/2777006
114. Feng Y (2020) A new deep-neural-network-based missing transverse momentum estimator, and its application to W recoil. PhD thesis, University of Maryland, College Park, https://doi.org/10.13016/e6ze-zycc
115. CMS Collaboration (2022) Identification of hadronic tau lepton decays using a deep neural network. JINST 17:P07023. https://doi.org/10.1088/1748-0221/17/07/P07023
116. NVIDIA Corporation (2020) NVIDIA T4 70W low profile PCIe GPU accelerator. NVIDIA Corporation, Santa Clara
117. Holzman B et al (2017) HEPCloud, a new paradigm for HEP facilities: CMS amazon web services investigation. Comput Softw Big Sci 1:1. https://doi.org/10.1007/s41781-017-0001-9. arXiv:1710.00100
118. Corporation Intel (2023) Intel 64 and IA-32 architectures software developer's manual. Intel Corporation, Santa Clara
119. SchedMD Slurm workload manager. https://slurm.schedmd.com/documentation.html. Accessed 08 Nov 2023
120. Inc Advanced Micro Devices (2020) AMD EPYC 7002 series processors power electronic health record solutions. Advanced Micro Devices Inc, Santa Clara

The CMS Collaboration

Yerevan Physics Institute, Yerevan, Armenia
A. Hayrapetyan, A. Tumasyan[1]

Institut für Hochenergiephysik, Vienna, Austria
W. Adam, J. W. Andrejkovic, T. Bergauer, S. Chatterjee, K. Damanakis, M. Dragicevic, P. S. Hussain, M. Jeitler[2], N. Krammer, A. Li, D. Liko, I. Mikulec, J. Schieck[2], R. Schöfbeck, D. Schwarz, M. Sonawane, S. Templ, W. Waltenberger, C.-E. Wulz[2]

Universiteit Antwerpen, Antwerpen, Belgium
M. R. Darwish[3], T. Janssen, P. Van Mechelen

Vrije Universiteit Brussel, Brussel, Belgium
E. S. Bols, J. D'Hondt, S. Dansana, A. De Moor, M. Delcourt, H. El Faham, S. Lowette, I. Makarenko, D. Müller, A. R. Sahasransu, S. Tavernier, M. Tytgat[4], G. P. Van Onsem, S. Van Putte, D. Vannerom

Université Libre de Bruxelles, Bruxelles, Belgium
B. Clerbaux, A. K. Das, G. De Lentdecker, L. Favart, P. Gianneios, D. Hohov, J. Jaramillo, A. Khalilzadeh, F. A. Khan, K. Lee, M. Mahdavikhorrami, A. Malara, S. Paredes, L. Thomas, M. Vanden Bemden, C. Vander Velde, P. Vanlaer

Ghent University, Ghent, Belgium
M. De Coen, D. Dobur, Y. Hong, J. Knolle, L. Lambrecht, G. Mestdach, K. Mota Amarilo, C. Rendón, A. Samalan, K. Skovpen, N. Van Den Bossche, J. van der Linden, L. Wezenbeek

Université Catholique de Louvain, Louvain-la-Neuve, Belgium
A. Benecke, A. Bethani, G. Bruno, C. Caputo, C. Delaere, I. S. Donertas, A. Giammanco, K. Jaffel, Sa. Jain, V. Lemaitre, J. Lidrych, P. Mastrapasqua, K. Mondal, T. T. Tran, S. Wertz

Centro Brasileiro de Pesquisas Fisicas, Rio de Janeiro, Brazil
G. A. Alves, E. Coelho, C. Hensel, T. Menezes De Oliveira, A. Moraes, P. Rebello Teles, M. Soeiro

Universidade do Estado do Rio de Janeiro, Rio de Janeiro, Brazil
W. L. Aldá Júnior, M. Alves Gallo Pereira, M. Barroso Ferreira Filho, H. Brandao Malbouisson, W. Carvalho, J. Chinellato[5], E. M. Da Costa, G. G. Da Silveira[6], D. De Jesus Damiao, S. Fonseca De Souza, R. Gomes De Souza, J. Martins[7], C. Mora Herrera, L. Mundim, H. Nogima, J. P. Pinheiro, A. Santoro, A. Sznajder, M. Thiel, A. Vilela Pereira

Universidade Estadual Paulista, Universidade Federal do ABC, São Paulo, Brazil
C. A. Bernardes[6], L. Calligaris, T. R. Fernandez Perez Tomei, E. M. Gregores, P. G. Mercadante, S. F. Novaes, B. Orzari, Sandra S. Padula

Institute for Nuclear Research and Nuclear Energy, Bulgarian Academy of Sciences, Sofia, Bulgaria
A. Aleksandrov, G. Antchev, R. Hadjiiska, P. Iaydjiev, M. Misheva, M. Shopova, G. Sultanov

University of Sofia, Sofia, Bulgaria
A. Dimitrov, L. Litov, B. Pavlov, P. Petkov, A. Petrov, E. Shumka

Instituto De Alta Investigación, Universidad de Tarapacá, Casilla 7 D, Arica, Chile
S. Keshri, S. Thakur

Beihang University, Beijing, China
T. Cheng, T. Javaid, L. Yuan

Department of Physics, Tsinghua University, Beijing, China
Z. Hu, J. Liu, K. Yi[8,9]

Institute of High Energy Physics, Beijing, China
G. M. Chen[10], H. S. Chen[10], M. Chen[10], F. Iemmi, C. H. Jiang, A. Kapoor[11], H. Liao, Z.-A. Liu[12], R. Sharma[13], J. N. Song[12], J. Tao, C. Wang[10], J. Wang, Z. Wang[10], H. Zhang

State Key Laboratory of Nuclear Physics and Technology, Peking University, Beijing, China
A. Agapitos, Y. Ban, A. Levin, C. Li, Q. Li, Y. Mao, S. J. Qian, X. Sun, D. Wang, H. Yang, L. Zhang, C. Zhou

Sun Yat-sen University, Guangzhou, China
Z. You

University of Science and Technology of China, Hefei, China
N. Lu

Nanjing Normal University, Nanjing, China
G. Bauer[14]

Institute of Modern Physics and Key Laboratory of Nuclear Physics and Ion-beam Application (MOE) - Fudan University, Shanghai, China
X. Gao[15], D. Leggat, H. Okawa

Zhejiang University, Hangzhou, Zhejiang, China
Z. Lin, C. Lu, M. Xiao

Universidad de Los Andes, Bogota, Colombia
C. Avila, D. A. Barbosa Trujillo, A. Cabrera, C. Florez, J. Fraga, J. A. Reyes Vega

Universidad de Antioquia, Medellin, Colombia
J. Mejia Guisao, F. Ramirez, M. Rodriguez, J. D. Ruiz Alvarez

University of Split, Faculty of Electrical Engineering, Mechanical Engineering and Naval Architecture, Split, Croatia
D. Giljanovic, N. Godinovic, D. Lelas, A. Sculac

Faculty of Science, University of Split, Split, Croatia
M. Kovac, T. Sculac

Institute Rudjer Boskovic, Zagreb, Croatia
P. Bargassa, V. Brigljevic, B. K. Chitroda, D. Ferencek, S. Mishra, A. Starodumov[16], T. Susa

University of Cyprus, Nicosia, Cyprus
A. Attikis, K. Christoforou, S. Konstantinou, J. Mousa, C. Nicolaou, F. Ptochos, P. A. Razis, H. Rykaczewski, H. Saka, A. Stepennov

Charles University, Prague, Czech Republic
M. Finger, M. Finger Jr., A. Kveton

Escuela Politecnica Nacional, Quito, Ecuador
E. Ayala

Universidad San Francisco de Quito, Quito, Ecuador
E. Carrera Jarrin

Academy of Scientific Research and Technology of the Arab Republic of Egypt, Egyptian Network of High Energy Physics, Cairo, Egypt
A. A. Abdelalim[17,18], E. Salama[19,20]

Center for High Energy Physics (CHEP-FU), Fayoum University, El-Fayoum, Egypt
M. A. Mahmoud, Y. Mohammed

National Institute of Chemical Physics and Biophysics, Tallinn, Estonia
K. Ehataht, M. Kadastik, T. Lange, S. Nandan, C. Nielsen, J. Pata, M. Raidal, L. Tani, C. Veelken

Department of Physics, University of Helsinki, Helsinki, Finland
H. Kirschenmann, K. Osterberg, M. Voutilainen

Helsinki Institute of Physics, Helsinki, Finland
S. Bharthuar, E. Brücken, F. Garcia, K. T. S. Kallonen, R. Kinnunen, T. Lampén, K. Lassila-Perini, S. Lehti, T. Lindén, L. Martikainen, M. Myllymäki, M.m. Rantanen, H. Siikonen, E. Tuominen, J. Tuominiemi

Lappeenranta-Lahti University of Technology, Lappeenranta, Finland
P. Luukka, H. Petrow

IRFU, CEA, Université Paris-Saclay, Gif-sur-Yvette, France
M. Besancon, F. Couderc, M. Dejardin, D. Denegri, J. L. Faure, F. Ferri, S. Ganjour, P. Gras, G. Hamel de Monchenault, V. Lohezic, J. Malcles, J. Rander, A. Rosowsky, M. Ö. Sahin, A. Savoy-Navarro[21], P. Simkina, M. Titov, M. Tornago

Laboratoire Leprince-Ringuet, CNRS/IN2P3, Ecole Polytechnique, Institut Polytechnique de Paris, Palaiseau, France
C. Baldenegro Barrera, F. Beaudette, A. Buchot Perraguin, P. Busson, A. Cappati, C. Charlot, M. Chiusi, F. Damas, O. Davignon, A. De Wit, B. A. Fontana Santos Alves, S. Ghosh, A. Gilbert, R. Granier de Cassagnac, A. Hakimi, B. Harikrishnan, L. Kalipoliti, G. Liu, J. Motta, M. Nguyen, C. Ochando, L. Portales, R. Salerno, J. B. Sauvan, Y. Sirois, A. Tarabini, E. Vernazza, A. Zabi, A. Zghiche

Université de Strasbourg, CNRS, IPHC UMR 7178, Strasbourg, France
J.-L. Agram[22], J. Andrea, D. Apparu, D. Bloch, J.-M. Brom, E. C. Chabert, C. Collard, S. Falke, U. Goerlach, C. Grimault, R. Haeberle, A.-C. Le Bihan, M. Meena, G. Saha, M. A. Sessini, P. Van Hove

Institut de Physique des 2 Infinis de Lyon (IP2I), Villeurbanne, France
S. Beauceron, B. Blancon, G. Boudoul, N. Chanon, J. Choi, D. Contardo, P. Depasse, C. Dozen[23], H. El Mamouni, J. Fay, S. Gascon, M. Gouzevitch, C. Greenberg, G. Grenier, B. Ille, I. B. Laktineh, M. Lethuillier, L. Mirabito, S. Perries, A. Purohit, M. Vander Donckt, P. Verdier, J. Xiao

Georgian Technical University, Tbilisi, Georgia
I. Bagaturia[24], I. Lomidze, Z. Tsamalaidze[16]

RWTH Aachen University, I. Physikalisches Institut, Aachen, Germany
V. Botta, L. Feld, K. Klein, M. Lipinski, D. Meuser, A. Pauls, N. Röwert, M. Teroerde

RWTH Aachen University, III. Physikalisches Institut A, Aachen, Germany
S. Diekmann, A. Dodonova, N. Eich, D. Eliseev, F. Engelke, J. Erdmann, M. Erdmann, P. Fackeldey, B. Fischer, T. Hebbeker, K. Hoepfner, F. Ivone, A. Jung, M.y. Lee, F. Mausolf, M. Merschmeyer, A. Meyer, S. Mukherjee, D. Noll, F. Nowotny, A. Pozdnyakov, Y. Rath, W. Redjeb, F. Rehm, H. Reithler, U. Sarkar, V. Sarkisovi, A. Schmidt, A. Sharma, J. L. Spah, A. Stein, F. Torres Da Silva De Araujo[25], L. Vigilante, S. Wiedenbeck, S. Zaleski

RWTH Aachen University, III. Physikalisches Institut B, Aachen, Germany
C. Dziwok, G. Flügge, W. Haj Ahmad[26], T. Kress, A. Nowack, O. Pooth, A. Stahl, T. Ziemons, A. Zotz

Deutsches Elektronen-Synchrotron, Hamburg, Germany
H. Aarup Petersen, M. Aldaya Martin, J. Alimena, S. Amoroso, Y. An, S. Baxter, M. Bayatmakou, H. Becerril Gonzalez, O. Behnke, A. Belvedere, S. Bhattacharya, F. Blekman[27], K. Borras[28], A. Campbell, A. Cardini, C. Cheng, F. Colombina, S. Consuegra Rodríguez, G. Correia Silva, M. De Silva, G. Eckerlin, D. Eckstein, L. I. Estevez Banos, O. Filatov, E. Gallo[27], A. Geiser, A. Giraldi, V. Guglielmi, M. Guthoff, A. Hinzmann, A. Jafari[29], L. Jeppe, N. Z. Jomhari, B. Kaech, M. Kasemann, C. Kleinwort, R. Kogler, M. Komm, D. Krücker, W. Lange, D. Leyva Pernia, K. Lipka[30], W. Lohmann[31], R. Mankel, I.-A. Melzer-Pellmann, M. Mendizabal Morentin, A. B. Meyer, G. Milella, A. Mussgiller,

L. P. Nair, A. Nürnberg, Y. Otarid, J. Park, D. Pérez Adán, E. Ranken, A. Raspereza, B. Ribeiro Lopes, J. Rübenach, A. Saggio, M. Scham[32,28], S. Schnake[28], P. Schütze, C. Schwanenberger[27], D. Selivanova, K. Sharko, M. Shchedrolosiev, R. E. Sosa Ricardo, D. Stafford, F. Vazzoler, A. Ventura Barroso, R. Walsh, Q. Wang, Y. Wen, K. Wichmann, L. Wiens[28], C. Wissing, Y. Yang, A. Zimermmane Castro Santos

University of Hamburg, Hamburg, Germany
A. Albrecht, S. Albrecht, M. Antonello, S. Bein, L. Benato, S. Bollweg, M. Bonanomi, P. Connor, M. Eich, K. El Morabit, Y. Fischer, C. Garbers, E. Garutti, A. Grohsjean, J. Haller, H. R. Jabusch, G. Kasieczka, P. Keicher, R. Klanner, W. Korcari, T. Kramer, V. Kutzner, F. Labe, J. Lange, A. Lobanov, C. Matthies, A. Mehta, L. Moureaux, M. Mrowietz, A. Nigamova, Y. Nissan, A. Paasch, K. J. Pena Rodriguez, T. Quadfasel, B. Raciti, M. Rieger, D. Savoiu, J. Schindler, P. Schleper, M. Schröder, J. Schwandt, M. Sommerhalder, H. Stadie, G. Steinbrück, A. Tews, M. Wolf

Karlsruher Institut fuer Technologie, Karlsruhe, Germany
S. Brommer, M. Burkart, E. Butz, T. Chwalek, A. Dierlamm, A. Droll, N. Faltermann, M. Giffels, A. Gottmann, F. Hartmann[33], R. Hofsaess, M. Horzela, U. Husemann, J. Kieseler, M. Klute, R. Koppenhöfer, J. M. Lawhorn, M. Link, A. Lintuluoto, S. Maier, S. Mitra, M. Mormile, Th. Müller, M. Neukum, M. Oh, M. Presilla, G. Quast, K. Rabbertz, B. Regnery, N. Shadskiy, I. Shvetsov, H. J. Simonis, M. Toms, N. Trevisani, R. F. Von Cube, M. Wassmer, S. Wieland, F. Wittig, R. Wolf, X. Zuo

Institute of Nuclear and Particle Physics (INPP), NCSR Demokritos, Aghia Paraskevi, Greece
G. Anagnostou, G. Daskalakis, A. Kyriakis, A. Papadopoulos[33], A. Stakia

National and Kapodistrian University of Athens, Athens, Greece
P. Kontaxakis, G. Melachroinos, A. Panagiotou, I. Papavergou, I. Paraskevas, N. Saoulidou, K. Theofilatos, E. Tziaferi, K. Vellidis, I. Zisopoulos

National Technical University of Athens, Athens, Greece
G. Bakas, T. Chatzistavrou, G. Karapostoli, K. Kousouris, I. Papakrivopoulos, E. Siamarkou, G. Tsipolitis, A. Zacharopoulou

University of Ioánnina, Ioánnina, Greece
K. Adamidis, I. Bestintzanos, I. Evangelou, C. Foudas, C. Kamtsikis, P. Katsoulis, P. Kokkas, P. G. Kosmoglou Kioseoglou, N. Manthos, I. Papadopoulos, J. Strologas

HUN-REN Wigner Research Centre for Physics, Budapest, Hungary
M. Bartók[34], C. Hajdu, D. Horvath[35,36], K. Márton, F. Sikler, V. Veszpremi

MTA-ELTE Lendület CMS Particle and Nuclear Physics Group, Eötvös Loránd University, Budapest, Hungary
M. Csanád, K. Farkas, M. M. A. Gadallah[37], Á. Kadlecsik, P. Major, K. Mandal, G. Pásztor, A. J. Rádl[38], G. I. Veres

Faculty of Informatics, University of Debrecen, Debrecen, Hungary
P. Raics, B. Ujvari, G. Zilizi

Institute of Nuclear Research ATOMKI, Debrecen, Hungary
G. Bencze, S. Czellar, J. Molnar, Z. Szillasi

Karoly Robert Campus, MATE Institute of Technology, Gyongyos, Hungary
T. Csorgo[38], F. Nemes[38], T. Novak

Panjab University, Chandigarh, India
J. Babbar, S. Bansal, S. B. Beri, V. Bhatnagar, G. Chaudhary, S. Chauhan, N. Dhingra[39], A. Kaur, A. Kaur, H. Kaur, M. Kaur, S. Kumar, K. Sandeep, T. Sheokand, J. B. Singh, A. Singla

University of Delhi, Delhi, India
A. Ahmed, A. Bhardwaj, A. Chhetri, B. C. Choudhary, A. Kumar, A. Kumar, M. Naimuddin, K. Ranjan, S. Saumya

Saha Institute of Nuclear Physics, HBNI, Kolkata, India
S. Baradia, S. Barman[40], S. Bhattacharya, S. Dutta, S. Dutta, S. Sarkar

Indian Institute of Technology Madras, Madras, India
M. M. Ameen, P. K. Behera, S. C. Behera, S. Chatterjee, P. Jana, P. Kalbhor, J. R. Komaragiri[41], D. Kumar[41], P. R. Pujahari, N. R. Saha, A. Sharma, A. K. Sikdar, S. Verma

Tata Institute of Fundamental Research-A, Mumbai, India
S. Dugad, M. Kumar, G. B. Mohanty, P. Suryadevara

Tata Institute of Fundamental Research-B, Mumbai, India
A. Bala, S. Banerjee, R. M. Chatterjee, R. K. Dewanjee[42], M. Guchait, Sh. Jain, A. Jaiswal, S. Karmakar, S. Kumar, G. Majumder, K. Mazumdar, S. Parolia, A. Thachayath

National Institute of Science Education and Research, An OCC of Homi Bhabha National Institute, Bhubaneswar, Odisha, India
S. Bahinipati[43], C. Kar, D. Maity[44], P. Mal, T. Mishra, V. K. Muraleedharan Nair Bindhu[44], K. Naskar[44], A. Nayak[44], P. Sadangi, P. Saha, S. K. Swain, S. Varghese[44], D. Vats[44]

Indian Institute of Science Education and Research (IISER), Pune, India
S. Acharya[45], A. Alpana, S. Dube, B. Gomber[45], B. Kansal, A. Laha, B. Sahu[45], S. Sharma, K. Y. Vaish

Isfahan University of Technology, Isfahan, Iran
H. Bakhshiansohi[46], E. Khazaie[47], M. Zeinali[48]

Institute for Research in Fundamental Sciences (IPM), Tehran, Iran
S. Chenarani[49], S. M. Etesami, M. Khakzad, M. Mohammadi Najafabadi

University College Dublin, Dublin, Ireland
M. Grunewald

INFN Sezione di Bari[a], Università di Bari[b], Politecnico di Bari[c], Bari, Italy
M. Abbrescia[a,b], R. Aly[a,c,17], A. Colaleo[a,b], D. Creanza[a,c], B. D'Anzi[a,b], N. De Filippis[a,c], M. De Palma[a,b], A. Di Florio[a,c], W. Elmetenawee[a,b,17], L. Fiore[a], G. Iaselli[a,c], M. Louka[a,b], G. Maggi[a,c], M. Maggi[a], I. Margjeka[a,b], V. Mastrapasqua[a,b], S. My[a,b], S. Nuzzo[a,b], A. Pellecchia[a,b], A. Pompili[a,b], G. Pugliese[a,c], R. Radogna[a], G. Ramirez-Sanchez[a,c], D. Ramos[a], A. Ranieri[a], L. Silvestris[a], F. M. Simone[a,b], Ü. Sözbilir[a], A. Stamerra[a], R. Venditti[a], P. Verwilligen[a], A. Zaza[a,b]

INFN Sezione di Bologna[a], Università di Bologna[b], Bologna, Italy
G. Abbiendi[a], C. Battilana[a,b], D. Bonacorsi[a,b], L. Borgonovi[a], R. Campanini[a,b], P. Capiluppi[a,b], A. Castro[a,b], F. R. Cavallo[a], M. Cuffiani[a,b], G. M. Dallavalle[a], T. Diotalevi[a,b], A. Fanfani[a,b], D. Fasanella[a,b], P. Giacomelli[a], L. Giommi[a,b], C. Grandi[a], L. Guiducci[a,b], S. Lo Meo[a,50], L. Lunerti[a,b], S. Marcellini[a], G. Masetti[a], F. L. Navarria[a,b], A. Perrotta[a], F. Primavera[a,b], A. M. Rossi[a,b], T. Rovelli[a,b], G. P. Siroli[a,b]

INFN Sezione di Catania[a], Università di Catania[b], Catania, Italy
S. Costa[a,b,51], A. Di Mattia[a], R. Potenza[a,b], A. Tricomi[a,b,51], C. Tuve[a,b]

INFN Sezione di Firenze[a], Università di Firenze[b], Firenze, Italy
P. Assiouras[a], G. Barbagli[a], G. Bardelli[a,b], B. Camaiani[a,b], A. Cassese[a], R. Ceccarelli[a], V. Ciulli[a,b], C. Civinini[a], R. D'Alessandro[a,b], E. Focardi[a,b], T. Kello[a], G. Latino[a,b], P. Lenzi[a,b], M. Lizzo[a], M. Meschini[a], S. Paoletti[a], A. Papanastassiou[a,b], G. Sguazzoni[a], L. Viliani[a]

INFN Laboratori Nazionali di Frascati, Frascati, Italy
L. Benussi, S. Bianco, S. Meola[52], D. Piccolo

INFN Sezione di Genova[a], Università di Genova[b], Genova, Italy
P. Chatagnon[a], F. Ferro[a], E. Robutti[a], S. Tosi[a,b]

INFN Sezione di Milano-Bicocca[a], Università di Milano-Bicocca[b], Milano, Italy
A. Benaglia[a], G. Boldrini[a,b], F. Brivio[a], F. Cetorelli[a], F. De Guio[a,b], M. E. Dinardo[a,b], P. Dini[a], S. Gennai[a], R. Gerosa[a,b], A. Ghezzi[a,b], P. Govoni[a,b], L. Guzzi[a], M. T. Lucchini[a,b], M. Malberti[a], S. Malvezzi[a], A. Massironi[a], D. Menasce[a], L. Moroni[a], M. Paganoni[a,b], D. Pedrini[a], B. S. Pinolini[a], S. Ragazzi[a,b], T. Tabarelli de Fatis[a,b], D. Zuolo[a]

INFN Sezione di Napoli[a], Università di Napoli 'Federico II'[b], Napoli, Italy; Università della Basilicata[c], Potenza, Italy; Scuola Superiore Meridionale (SSM)[d], Napoli, Italy
S. Buontempo[a], A. Cagnotta[a,b], F. Carnevali[a,b], N. Cavallo[a,c], F. Fabozzi[a,c], A. O. M. Iorio[a,b], L. Lista[a,b,53], P. Paolucci[a,33], B. Rossi[a], C. Sciacca[a,b]

INFN Sezione di Padova[a], Università di Padova[b], Padova, Italy; Università di Trento[c], Trento, Italy
R. Ardino[a], P. Azzi[a], N. Bacchetta[a,54], D. Bisello[a,b], P. Bortignon[a], A. Bragagnolo[a,b], P. Checchia[a], T. Dorigo[a], U. Gasparini[a,b], E. Lusiani[a], M. Margoni[a,b], F. Marini[a], A. T. Meneguzzo[a,b], M. Migliorini[a,b], M. Passaseo[a], J. Pazzini[a,b], P. Ronchese[a,b], R. Rossin[a,b], M. Sgaravatto[a], F. Simonetto[a,b], G. Strong[a], M. Tosi[a,b], A. Triossi[a,b], S. Ventura[a], H. Yarar[a,b], M. Zanetti[a,b], P. Zotto[a,b], A. Zucchetta[a,b], G. Zumerle[a,b]

INFN Sezione di Pavia[a], Università di Pavia[b], Pavia, Italy
S. Abu Zeid[a,20], C. Aimè[a,b], A. Braghieri[a], S. Calzaferri[a], D. Fiorina[a], P. Montagna[a,b], V. Re[a], C. Riccardi[a,b], P. Salvini[a], I. Vai[a,b], P. Vitulo[a,b]

INFN Sezione di Perugia[a], Università di Perugia[b], Perugia, Italy
S. Ajmal[a,b], G. M. Bilei[a], D. Ciangottini[a,b], L. Fanò[a,b], M. Magherini[a,b], G. Mantovani[a,b], V. Mariani[a,b], M. Menichelli[a], F. Moscatelli[a,55], A. Rossi[a,b], A. Santocchia[a,b], D. Spiga[a], T. Tedeschi[a,b]

INFN Sezione di Pisa[a], Università di Pisa[b], Scuola Normale Superiore di Pisa[c], Pisa, Italy; Università di Siena[d], Siena, Italy
P. Asenov[a,b], P. Azzurri[a], G. Bagliesi[a], R. Bhattacharya[a], L. Bianchini[a,b], T. Boccali[a], E. Bossini[a], D. Bruschini[a,c], R. Castaldi[a], M. A. Ciocci[a,b], M. Cipriani[a,b], V. D'Amante[a,d], R. Dell'Orso[a], S. Donato[a], A. Giassi[a], F. Ligabue[a,c], D. Matos Figueiredo[a], A. Messineo[a,b], M. Musich[a,b], F. Palla[a], A. Rizzi[a,b], G. Rolandi[a,c], S. Roy Chowdhury[a], T. Sarkar[a], A. Scribano[a], P. Spagnolo[a], R. Tenchini[a], G. Tonelli[a,b], N. Turini[a,d], A. Venturi[a], P. G. Verdini[a]

INFN Sezione di Roma[a], Sapienza Università di Roma[b], Roma, Italy
P. Barria[a], C. Basile[a,b], M. Campana[a,b], F. Cavallari[a], L. Cunqueiro Mendez[a,b], D. Del Re[a,b], E. Di Marco[a], M. Diemoz[a], F. Errico[a,b], E. Longo[a,b], P. Meridiani[a], J. Mijuskovic[a,b], G. Organtini[a,b], F. Pandolfi[a], R. Paramatti[a,b], C. Quaranta[a,b], S. Rahatlou[a,b], C. Rovelli[a], F. Santanastasio[a,b], L. Soffi[a]

INFN Sezione di Torino[a], Università di Torino[b], Torino, Italy; Università del Piemonte Orientale[c], Novara, Italy
N. Amapane[a,b], R. Arcidiacono[a,c], S. Argiro[a,b], M. Arneodo[a,c], N. Bartosik[a], R. Bellan[a,b], A. Bellora[a,b], C. Biino[a], C. Borca[a,b], N. Cartiglia[a], M. Costa[a,b], R. Covarelli[a,b], N. Demaria[a], L. Finco[a], M. Grippo[a,b], B. Kiani[a,b], F. Legger[a], F. Luongo[a,b], C. Mariotti[a], L. Markovic[a,b], S. Maselli[a], A. Mecca[a,b], E. Migliore[a,b], M. Monteno[a], R. Mulargia[a], M. M. Obertino[a,b], G. Ortona[a], L. Pacher[a,b], N. Pastrone[a], M. Pelliccioni[a], M. Ruspa[a,c], F. Siviero[a,b], V. Sola[a,b], A. Solano[a,b], A. Staiano[a], C. Tarricone[a,b], D. Trocino[a], G. Umoret[a,b], E. Vlasov[a,b]

INFN Sezione di Trieste[a], Università di Trieste[b], Trieste, Italy
S. Belforte[a], V. Candelise[a,b], M. Casarsa[a], F. Cossutti[a], K. De Leo[a,b], G. Della Ricca[a,b]

Kyungpook National University, Daegu, Korea
S. Dogra, J. Hong, C. Huh, B. Kim, D. H. Kim, J. Kim, H. Lee, S. W. Lee, C. S. Moon, Y. D. Oh, M. S. Ryu, S. Sekmen, Y. C. Yang

Department of Mathematics and Physics, GWNU, Gangneung, Korea
M. S. Kim

Chonnam National University, Institute for Universe and Elementary Particles, Kwangju, Korea
G. Bak, P. Gwak, H. Kim, D. H. Moon

Hanyang University, Seoul, Korea
E. Asilar, D. Kim, T. J. Kim, J. A. Merlin

Korea University, Seoul, Korea
S. Choi, S. Han, B. Hong, K. Lee, K. S. Lee, S. Lee, J. Park, S. K. Park, J. Yoo

Department of Physics, Kyung Hee University, Seoul, Korea
J. Goh, S. Yang

Sejong University, Seoul, Korea
H. S. Kim, Y. Kim, S. Lee

Seoul National University, Seoul, Korea
J. Almond, J. H. Bhyun, J. Choi, W. Jun, J. Kim, S. Ko, H. Kwon, H. Lee, J. Lee, J. Lee, B. H. Oh, S. B. Oh, H. Seo, U. K. Yang, I. Yoon

University of Seoul, Seoul, Korea
W. Jang, D. Y. Kang, Y. Kang, S. Kim, B. Ko, J. S. H. Lee, Y. Lee, I. C. Park, Y. Roh, I. J. Watson

Department of Physics, Yonsei University, Seoul, Korea
S. Ha, H. D. Yoo

Sungkyunkwan University, Suwon, Korea
M. Choi, M. R. Kim, H. Lee, Y. Lee, I. Yu

College of Engineering and Technology, American University of the Middle East (AUM), Dasman, Kuwait
T. Beyrouthy, Y. Maghrbi

Riga Technical University, Riga, Latvia
K. Dreimanis, A. Gaile, G. Pikurs, A. Potrebko, M. Seidel, V. Veckalns[56]

University of Latvia (LU), Riga, Latvia
N. R. Strautnieks

Vilnius University, Vilnius, Lithuania
M. Ambrozas, A. Juodagalvis, A. Rinkevicius, G. Tamulaitis

National Centre for Particle Physics, Universiti Malaya, Kuala Lumpur, Malaysia
N. Bin Norjoharuddeen, I. Yusuff[57], Z. Zolkapli

Universidad de Sonora (UNISON), Hermosillo, Mexico
J. F. Benitez, A. Castaneda Hernandez, H. A. Encinas Acosta, L. G. Gallegos Maríñez, M. León Coello, J. A. Murillo Quijada, A. Sehrawat, L. Valencia Palomo

Centro de Investigacion y de Estudios Avanzados del IPN, Mexico City, Mexico
G. Ayala, H. Castilla-Valdez, H. Crotte Ledesma, E. De La Cruz-Burelo, I. Heredia-De La Cruz[58], R. Lopez-Fernandez, C. A. Mondragon Herrera, A. Sánchez Hernández

Universidad Iberoamericana, Mexico City, Mexico
C. Oropeza Barrera, M. Ramírez García

Benemerita Universidad Autonoma de Puebla, Puebla, Mexico
I. Bautista, I. Pedraza, H. A. Salazar Ibarguen, C. Uribe Estrada

University of Montenegro, Podgorica, Montenegro
I. Bubanja, N. Raicevic

University of Canterbury, Christchurch, New Zealand
P. H. Butler

National Centre for Physics, Quaid-I-Azam University, Islamabad, Pakistan
A. Ahmad, M. I. Asghar, A. Awais, M. I. M. Awan, H. R. Hoorani, W. A. Khan

AGH University of Krakow, Faculty of Computer Science, Electronics and Telecommunications, Krakow, Poland
V. Avati, L. Grzanka, M. Malawski

National Centre for Nuclear Research, Swierk, Poland
H. Bialkowska, M. Bluj, B. Boimska, M. Górski, M. Kazana, M. Szleper, P. Zalewski

Institute of Experimental Physics, Faculty of Physics, University of Warsaw, Warsaw, Poland
K. Bunkowski, K. Doroba, A. Kalinowski, M. Konecki, J. Krolikowski, A. Muhammad

Warsaw University of Technology, Warsaw, Poland
K. Pozniak, W. Zabolotny

Laboratório de Instrumentação e Física Experimental de Partículas, Lisboa, Portugal
M. Araujo, D. Bastos, C. Beirão Da Cruz E Silva, A. Boletti, M. Bozzo, T. Camporesi, G. Da Molin, P. Faccioli, M. Gallinaro, J. Hollar, N. Leonardo, T. Niknejad, A. Petrilli, M. Pisano, J. Seixas, J. Varela, J. W. Wulff

Faculty of Physics, University of Belgrade, Belgrade, Serbia
P. Adzic, P. Milenovic

VINCA Institute of Nuclear Sciences, University of Belgrade, Belgrade, Serbia
M. Dordevic, J. Milosevic, V. Rekovic

Centro de Investigaciones Energéticas Medioambientales y Tecnológicas (CIEMAT), Madrid, Spain
M. Aguilar-Benitez, J. Alcaraz Maestre, Cristina F. Bedoya, M. Cepeda, M. Cerrada, N. Colino, B. De La Cruz, A. Delgado Peris, A. Escalante Del Valle, D. Fernández Del Val, J. P. Fernández Ramos, J. Flix, M. C. Fouz, O. Gonzalez Lopez, S. Goy Lopez, J. M. Hernandez, M. I. Josa, D. Moran, C. M. Morcillo Perez, Á. Navarro Tobar, C. Perez Dengra, A. Pérez-Calero Yzquierdo, J. Puerta Pelayo, I. Redondo, D. D. Redondo Ferrero, L. Romero, S. Sánchez Navas, L. Urda Gómez, J. Vazquez Escobar, C. Willmott

Universidad Autónoma de Madrid, Madrid, Spain
J. F. de Trocóniz

Universidad de Oviedo, Instituto Universitario de Ciencias y Tecnologías Espaciales de Asturias (ICTEA), Oviedo, Spain
B. Alvarez Gonzalez, J. Cuevas, J. Fernandez Menendez, S. Folgueras, I. Gonzalez Caballero, J. R. González Fernández, E. Palencia Cortezon, C. Ramón Álvarez, V. Rodríguez Bouza, A. Soto Rodríguez, A. Trapote, C. Vico Villalba, P. Vischia

Instituto de Física de Cantabria (IFCA), CSIC-Universidad de Cantabria, Santander, Spain
S. Bhowmik, S. Blanco Fernández, J. A. Brochero Cifuentes, I. J. Cabrillo, A. Calderon, J. Duarte Campderros, M. Fernandez, G. Gomez, C. Lasaosa García, C. Martinez Rivero, P. Martinez Ruiz del Arbol, F. Matorras, P. Matorras Cuevas, E. Navarrete Ramos, J. Piedra Gomez, L. Scodellaro, I. Vila, J. M. Vizan Garcia

University of Colombo, Colombo, Sri Lanka
M. K. Jayananda, B. Kailasapathy[59], D. U. J. Sonnadara, D. D. C. Wickramarathna

Department of Physics, University of Ruhuna, Matara, Sri Lanka
W. G. D. Dharmaratna[60], K. Liyanage, N. Perera, N. Wickramage

CERN, European Organization for Nuclear Research, Geneva, Switzerland
D. Abbaneo, C. Amendola, E. Auffray, G. Auzinger, J. Baechler, D. Barney, A. Bermúdez Martínez, M. Bianco, B. Bilin, A. A. Bin Anuar, A. Bocci, C. Botta, E. Brondolin, C. Caillol, G. Cerminara, N. Chernyavskaya, D. d'Enterria, A. Dabrowski, A. David, A. De Roeck, M. M. Defranchis, M. Deile, M. Dobson, L. Forthomme, G. Franzoni, W. Funk, S. Giani, D. Gigi, K. Gill, F. Glege, L. Gouskos,

M. Haranko, J. Hegeman, B. Huber, V. Innocente, T. James, P. Janot, S. Laurila, P. Lecoq, E. Leutgeb, C. Lourenço, B. Maier, L. Malgeri, M. Mannelli, A. C. Marini, M. Matthewman, F. Meijers, S. Mersi, E. Meschi, V. Milosevic, F. Monti, F. Moortgat, M. Mulders, I. Neutelings, S. Orfanelli, F. Pantaleo, G. Petrucciani, A. Pfeiffer, M. Pierini, D. Piparo, H. Qu, D. Rabady, G. Reales Gutiérrez, M. Rovere, H. Sakulin, S. Scarfi, C. Schwick, M. Selvaggi, A. Sharma, K. Shchelina, P. Silva, P. Sphicas[61], A. G. Stahl Leiton, A. Steen, S. Summers, D. Treille, P. Tropea, A. Tsirou, D. Walter, J. Wanczyk[62], J. Wang, S. Wuchterl, P. Zehetner, P. Zejdl, W. D. Zeuner

Paul Scherrer Institut, Villigen, Switzerland
T. Bevilacqua[63], L. Caminada[63], A. Ebrahimi, W. Erdmann, R. Horisberger, Q. Ingram, H. C. Kaestli, D. Kotlinski, C. Lange, M. Missiroli[63], L. Noehte[63], T. Rohe

ETH Zurich, Institute for Particle Physics and Astrophysics (IPA), Zurich, Switzerland
T. K. Aarrestad, K. Androsov[62], M. Backhaus, A. Calandri, C. Cazzaniga, K. Datta, A. De Cosa, G. Dissertori, M. Dittmar, M. Donegà, F. Eble, M. Galli, K. Gedia, F. Glessgen, C. Grab, D. Hits, W. Lustermann, A.-M. Lyon, R. A. Manzoni, M. Marchegiani, L. Marchese, C. Martin Perez, A. Mascellani[62], F. Nessi-Tedaldi, F. Pauss, V. Perovic, S. Pigazzini, C. Reissel, T. Reitenspiess, B. Ristic, F. Riti, R. Seidita, J. Steggemann[62], D. Valsecchi, R. Wallny

Universität Zürich, Zurich, Switzerland
C. Amsler[64], P. Bärtschi, D. Brzhechko, M. F. Canelli, K. Cormier, J. K. Heikkilä, M. Huwiler, W. Jin, A. Jofrehei, B. Kilminster, S. Leontsinis, S. P. Liechti, A. Macchiolo, P. Meiring, U. Molinatti, A. Reimers, P. Robmann, S. Sanchez Cruz, M. Senger, F. Stäger, Y. Takahashi, R. Tramontano

National Central University, Chung-Li, Taiwan
C. Adloff[65], D. Bhowmik, C. M. Kuo, W. Lin, P. K. Rout, P. C. Tiwari[41], S. S. Yu

National Taiwan University (NTU), Taipei, Taiwan
L. Ceard, Y. Chao, K. F. Chen, P.s. Chen, Z.g. Chen, A. De Iorio, W.-S. Hou, T.h. Hsu, Y.w. Kao, R. Khurana, G. Kole, Y.y. Li, R.-S. Lu, E. Paganis, X.f. Su, J. Thomas-Wilsker, L.s. Tsai, H.y. Wu, E. Yazgan

High Energy Physics Research Unit, Department of Physics, Faculty of Science, Chulalongkorn University, Bangkok, Thailand
C. Asawatangtrakuldee, N. Srimanobhas, V. Wachirapusitanand

Çukurova University, Physics Department, Science and Art Faculty, Adana, Turkey
D. Agyel, F. Boran, Z. S. Demiroglu, F. Dolek, I. Dumanoglu[66], E. Eskut, Y. Guler[67], E. Gurpinar Guler[67], C. Isik, O. Kara, A. Kayis Topaksu, U. Kiminsu, G. Onengut, K. Ozdemir[68], A. Polatoz, B. Tali[69], U. G. Tok, S. Turkcapar, E. Uslan, I. S. Zorbakir

Middle East Technical University, Physics Department, Ankara, Turkey
M. Yalvac[70]

Bogazici University, Istanbul, Turkey
B. Akgun, I. O. Atakisi, E. Gülmez, M. Kaya[71], O. Kaya[72], S. Tekten[73]

Istanbul Technical University, Istanbul, Turkey
A. Cakir, K. Cankocak[66,74], Y. Komurcu, S. Sen[75]

Istanbul University, Istanbul, Turkey
O. Aydilek, S. Cerci[69], V. Epshteyn, B. Hacisahinoglu, I. Hos[76], B. Kaynak, S. Ozkorucuklu, O. Potok, H. Sert, C. Simsek, C. Zorbilmez

Yildiz Technical University, Istanbul, Turkey
B. Isildak[77], D. Sunar Cerci[69]

Institute for Scintillation Materials of National Academy of Science of Ukraine, Kharkiv, Ukraine
A. Boyaryntsev, B. Grynyov

National Science Centre, Kharkiv Institute of Physics and Technology, Kharkiv, Ukraine
L. Levchuk

University of Bristol, Bristol, UK
D. Anthony, J. J. Brooke, A. Bundock, F. Bury, E. Clement, D. Cussans, H. Flacher, M. Glowacki, J. Goldstein, H. F. Heath, L. Kreczko, S. Paramesvaran, L. Robertshaw, S. Seif El Nasr-Storey, V. J. Smith, N. Stylianou[78], K. Walkingshaw Pass, R. White

Rutherford Appleton Laboratory, Didcot, UK
A. H. Ball, K. W. Bell, A. Belyaev[79], C. Brew, R. M. Brown, D. J. A. Cockerill, C. Cooke, K. V. Ellis, K. Harder, S. Harper, M.-L. Holmberg[80], J. Linacre, K. Manolopoulos, D. M. Newbold, E. Olaiya, D. Petyt, T. Reis, G. Salvi, T. Schuh, C. H. Shepherd-Themistocleous, I. R. Tomalin, T. Williams

Imperial College, London, UK
R. Bainbridge, P. Bloch, C. E. Brown, O. Buchmuller, V. Cacchio, C. A. Carrillo Montoya, G. S. Chahal[81], D. Colling, J. S. Dancu, I. Das, P. Dauncey, G. Davies, J. Davies, M. Della Negra, S. Fayer, G. Fedi, G. Hall, M. H. Hassanshahi, A. Howard, G. Iles, M. Knight, J. Langford, J. León Holgado, L. Lyons, A.-M. Magnan, S. Malik, M. Mieskolainen, J. Nash[82], M. Pesaresi, B. C. Radburn-Smith, A. Richards, A. Rose, K. Savva, C. Seez, R. Shukla, A. Tapper, K. Uchida, G. P. Uttley, L. H. Vage, T. Virdee[33], M. Vojinovic, N. Wardle, D. Winterbottom

Brunel University, Uxbridge, UK
K. Coldham, J. E. Cole, A. Khan, P. Kyberd, I. D. Reid

Baylor University, Waco, TX, USA
S. Abdullin, A. Brinkerhoff, B. Caraway, J. Dittmann, K. Hatakeyama, J. Hiltbrand, B. McMaster, M. Saunders, S. Sawant, C. Sutantawibul, J. Wilson

Catholic University of America, Washington, DC, USA
R. Bartek, A. Dominguez, C. Huerta Escamilla, A. E. Simsek, R. Uniyal, A. M. Vargas Hernandez

The University of Alabama, Tuscaloosa, AL, USA
B. Bam, R. Chudasama, S. I. Cooper, S. V. Gleyzer, C. U. Perez, P. Rumerio[83], E. Usai, R. Yi

Boston University, Boston, MA, USA
A. Akpinar, D. Arcaro, C. Cosby, Z. Demiragli, C. Erice, C. Fangmeier, C. Fernandez Madrazo, E. Fontanesi, D. Gastler, F. Golf, S. Jeon, I. Reed, J. Rohlf, K. Salyer, D. Sperka, D. Spitzbart, I. Suarez, A. Tsatsos, S. Yuan, A. G. Zecchinelli

Brown University, Providence, RI, USA
G. Benelli, X. Coubez[28], D. Cutts, M. Hadley, U. Heintz, J. M. Hogan[84], T. Kwon, G. Landsberg, K. T. Lau, D. Li, J. Luo, S. Mondal, M. Narain[†], N. Pervan, S. Sagir[85], F. Simpson, M. Stamenkovic, X. Yan, W. Zhang

University of California, Davis, CA, USA
S. Abbott, J. Bonilla, C. Brainerd, R. Breedon, M. Calderon De La Barca Sanchez, M. Chertok, M. Citron, J. Conway, P. T. Cox, R. Erbacher, F. Jensen, O. Kukral, G. Mocellin, M. Mulhearn, D. Pellett, W. Wei, Y. Yao, F. Zhang

University of California, Los Angeles, CA, USA
M. Bachtis, R. Cousins, A. Datta, G. Flores Avila, J. Hauser, M. Ignatenko, M. A. Iqbal, T. Lam, E. Manca, A. Nunez Del Prado, D. Saltzberg, V. Valuev

University of California, Riverside, CA, USA
R. Clare, J. W. Gary, M. Gordon, G. Hanson, W. Si, S. Wimpenny[†]

University of California, San Diego, La Jolla, CA, USA
J. G. Branson, S. Cittolin, S. Cooperstein, D. Diaz, J. Duarte, L. Giannini, J. Guiang, R. Kansal,

V. Krutelyov, R. Lee, J. Letts, M. Masciovecchio, F. Mokhtar, S. Mukherjee, M. Pieri, M. Quinnan, B. V. Sathia Narayanan, V. Sharma, M. Tadel, E. Vourliotis, F. Würthwein, Y. Xiang, A. Yagil

University of California, Santa Barbara, Department of Physics, Santa Barbara, CA, USA
A. Barzdukas, L. Brennan, C. Campagnari, J. Incandela, J. Kim, A. J. Li, P. Masterson, H. Mei, J. Richman, U. Sarica, R. Schmitz, F. Setti, J. Sheplock, D. Stuart, T. Á. Vámi, S. Wang

California Institute of Technology, Pasadena, CA, USA
A. Bornheim, O. Cerri, A. Latorre, J. Mao, H. B. Newman, M. Spiropulu, J. R. Vlimant, C. Wang, S. Xie, R. Y. Zhu

Carnegie Mellon University, Pittsburgh, PA, USA
J. Alison, S. An, M. B. Andrews, P. Bryant, M. Cremonesi, V. Dutta, T. Ferguson, A. Harilal, C. Liu, T. Mudholkar, S. Murthy, P. Palit, M. Paulini, A. Roberts, A. Sanchez, W. Terrill

University of Colorado Boulder, Boulder, CO, USA
J. P. Cumalat, W. T. Ford, A. Hart, A. Hassani, G. Karathanasis, E. MacDonald, N. Manganelli, A. Perloff, C. Savard, N. Schonbeck, K. Stenson, K. A. Ulmer, S. R. Wagner, N. Zipper

Cornell University, Ithaca, NY, USA
J. Alexander, S. Bright-Thonney, X. Chen, D. J. Cranshaw, J. Fan, X. Fan, D. Gadkari, S. Hogan, P. Kotamnives, J. Monroy, M. Oshiro, J. R. Patterson, J. Reichert, M. Reid, A. Ryd, J. Thom, P. Wittich, R. Zou

Fermi National Accelerator Laboratory, Batavia, IL, USA
M. Albrow, M. Alyari, O. Amram, G. Apollinari, A. Apresyan, L. A. T. Bauerdick, D. Berry, J. Berryhill, P. C. Bhat, K. Burkett, J. N. Butler, A. Canepa, G. B. Cerati, H. W. K. Cheung, F. Chlebana, G. Cummings, J. Dickinson, I. Dutta, V. D. Elvira, Y. Feng, J. Freeman, A. Gandrakota, Z. Gecse, L. Gray, D. Green, A. Grummer, S. Grünendahl, D. Guerrero, O. Gutsche, R. M. Harris, R. Heller, T. C. Herwig, J. Hirschauer, L. Horyn, B. Jayatilaka, S. Jindariani, M. Johnson, U. Joshi, T. Klijnsma, B. Klima, K. H. M. Kwok, S. Lammel, D. Lincoln, R. Lipton, T. Liu, C. Madrid, K. Maeshima, C. Mantilla, D. Mason, P. McBride, P. Merkel, S. Mrenna, S. Nahn, J. Ngadiuba, D. Noonan, V. Papadimitriou, N. Pastika, K. Pedro, C. Pena[86], F. Ravera, A. Reinsvold Hall[87], L. Ristori, E. Sexton-Kennedy, N. Smith, A. Soha, L. Spiegel, S. Stoynev, J. Strait, L. Taylor, S. Tkaczyk, N. V. Tran, L. Uplegger, E. W. Vaandering, I. Zoi

University of Florida, Gainesville, FL, USA
C. Aruta, P. Avery, D. Bourilkov, L. Cadamuro, P. Chang, V. Cherepanov, R. D. Field, E. Koenig, M. Kolosova, J. Konigsberg, A. Korytov, K. Matchev, N. Menendez, G. Mitselmakher, K. Mohrman, A. Muthirakalayil Madhu, N. Rawal, D. Rosenzweig, S. Rosenzweig, J. Wang

Florida State University, Tallahassee, FL, USA
T. Adams, A. Al Kadhim, A. Askew, S. Bower, R. Habibullah, V. Hagopian, R. Hashmi, R. S. Kim, S. Kim, T. Kolberg, G. Martinez, H. Prosper, P. R. Prova, M. Wulansatiti, R. Yohay, J. Zhang

Florida Institute of Technology, Melbourne, FL, USA
B. Alsufyani, M. M. Baarmand, S. Butalla, T. Elkafrawy[20], M. Hohlmann, R. Kumar Verma, M. Rahmani, E. Yanes

University of Illinois Chicago, Chicago, USA
M. R. Adams, A. Baty, C. Bennett, R. Cavanaugh, R. Escobar Franco, O. Evdokimov, C. E. Gerber, D. J. Hofman, J.h. Lee, D. S. Lemos, A. H. Merrit, C. Mills, S. Nanda, G. Oh, B. Ozek, D. Pilipovic, R. Pradhan, T. Roy, S. Rudrabhatla, M. B. Tonjes, N. Varelas, Z. Ye, J. Yoo

The University of Iowa, Iowa, USA
M. Alhusseini, D. Blend, K. Dilsiz[88], L. Emediato, G. Karaman, O. K. Köseyan, J.-P. Merlo, A. Mestvirishvili[89], J. Nachtman, O. Neogi, H. Ogul[90], Y. Onel, A. Penzo, C. Snyder, E. Tiras[91]

Johns Hopkins University, Baltimore, MD, USA
B. Blumenfeld, L. Corcodilos, J. Davis, A. V. Gritsan, L. Kang, S. Kyriacou, P. Maksimovic, M. Roguljic, J. Roskes, S. Sekhar, M. Swartz

The University of Kansas, Lawrence, KS, USA
A. Abreu, L. F. Alcerro Alcerro, J. Anguiano, P. Baringer, A. Bean, Z. Flowers, D. Grove, J. King, G. Krintiras, M. Lazarovits, C. Le Mahieu, J. Marquez, N. Minafra, M. Murray, M. Nickel, M. Pitt, S. Popescu[92], C. Rogan, C. Royon, R. Salvatico, S. Sanders, C. Smith, Q. Wang, G. Wilson

Kansas State University, Manhattan, KS, USA
B. Allmond, A. Ivanov, K. Kaadze, A. Kalogeropoulos, D. Kim, Y. Maravin, J. Natoli, D. Roy, G. Sorrentino

Lawrence Livermore National Laboratory, Livermore, CA, USA
F. Rebassoo, D. Wright

University of Maryland, College Park, MD, USA
A. Baden, A. Belloni, Y. M. Chen, S. C. Eno, N. J. Hadley, S. Jabeen, R. G. Kellogg, T. Koeth, Y. Lai, S. Lascio, A. C. Mignerey, S. Nabili, C. Palmer, C. Papageorgakis, M. M. Paranjpe, L. Wang

Massachusetts Institute of Technology, Cambridge, MA, USA
J. Bendavid, I. A. Cali, M. D'Alfonso, J. Eysermans, C. Freer, G. Gomez-Ceballos, M. Goncharov, G. Grosso, P. Harris, D. Hoang, D. Kovalskyi, J. Krupa, L. Lavezzo, Y.-J. Lee, K. Long, A. Novak, C. Paus, D. Rankin, C. Roland, G. Roland, S. Rothman, G. S. F. Stephans, Z. Wang, B. Wyslouch, T. J. Yang

University of Minnesota, Minneapolis, MN, USA
B. Crossman, B. M. Joshi, C. Kapsiak, M. Krohn, D. Mahon, J. Mans, B. Marzocchi, S. Pandey, M. Revering, R. Rusack, R. Saradhy, N. Schroeder, N. Strobbe, M. A. Wadud

University of Mississippi, Oxford, MS, USA
L. M. Cremaldi

University of Nebraska-Lincoln, Lincoln, NE, USA
K. Bloom, D. R. Claes, G. Haza, J. Hossain, C. Joo, I. Kravchenko, J. E. Siado, W. Tabb, A. Vagnerini, A. Wightman, F. Yan, D. Yu

State University of New York at Buffalo, Buffalo, NY, USA
H. Bandyopadhyay, L. Hay, I. Iashvili, A. Kharchilava, M. Morris, D. Nguyen, S. Rappoccio, H. Rejeb Sfar, A. Williams

Northeastern University, Boston, MA, USA
G. Alverson, E. Barberis, J. Dervan, Y. Haddad, Y. Han, A. Krishna, J. Li, M. Lu, G. Madigan, R. Mccarthy, D. M. Morse, V. Nguyen, T. Orimoto, A. Parker, L. Skinnari, B. Wang, D. Wood

Northwestern University, Evanston, IL, USA
S. Bhattacharya, J. Bueghly, Z. Chen, S. Dittmer, K. A. Hahn, Y. Liu, Y. Miao, D. G. Monk, M. H. Schmitt, A. Taliercio, M. Velasco

University of Notre Dame, Notre Dame, IN, USA
G. Agarwal, R. Band, R. Bucci, S. Castells, A. Das, R. Goldouzian, M. Hildreth, K. W. Ho, K. Hurtado Anampa, T. Ivanov, C. Jessop, K. Lannon, J. Lawrence, N. Loukas, L. Lutton, J. Mariano, N. Marinelli, I. Mcalister, T. McCauley, C. Mcgrady, C. Moore, Y. Musienko[16], H. Nelson, M. Osherson, A. Piccinelli, R. Ruchti, A. Townsend, Y. Wan, M. Wayne, H. Yockey, M. Zarucki, L. Zygala

The Ohio State University, Columbus, OH, USA
A. Basnet, B. Bylsma, M. Carrigan, L. S. Durkin, C. Hill, M. Joyce, M. Nunez Ornelas, K. Wei, B. L. Winer, B. R. Yates

Princeton University, Princeton, NJ, USA
F. M. Addesa, H. Bouchamaoui, P. Das, G. Dezoort, P. Elmer, A. Frankenthal, B. Greenberg, N. Haubrich, G. Kopp, S. Kwan, D. Lange, A. Loeliger, D. Marlow, I. Ojalvo, J. Olsen, A. Shevelev, D. Stickland, C. Tully

University of Puerto Rico, Mayaguez, PR, USA
S. Malik

Purdue University, West Lafayette, IN, USA
A. S. Bakshi, V. E. Barnes, S. Chandra, R. Chawla, S. Das, A. Gu, L. Gutay, M. Jones, A. W. Jung, D. Kondratyev, A. M. Koshy, M. Liu, G. Negro, N. Neumeister, G. Paspalaki, S. Piperov, V. Scheurer, J. F. Schulte, M. Stojanovic, J. Thieman, A. K. Virdi, F. Wang, W. Xie

Purdue University Northwest, Hammond, IN, USA
J. Dolen, N. Parashar, A. Pathak

Rice University, Houston, TX, USA
D. Acosta, T. Carnahan, K. M. Ecklund, P. J. Fernández Manteca, S. Freed, P. Gardner, F. J. M. Geurts, W. Li, O. Miguel Colin, B. P. Padley, R. Redjimi, J. Rotter, E. Yigitbasi, Y. Zhang

University of Rochester, Rochester, NY, USA
A. Bodek, P. de Barbaro, R. Demina, J. L. Dulemba, A. Garcia-Bellido, O. Hindrichs, A. Khukhunaishvili, N. Parmar, P. Parygin[93], E. Popova[93], R. Taus

The Rockefeller University, New York, NY, USA
K. Goulianos

Rutgers, The State University of New Jersey, Piscataway, NJ, USA
B. Chiarito, J. P. Chou, Y. Gershtein, E. Halkiadakis, M. Heindl, C. Houghton, D. Jaroslawski, O. Karacheban[31], I. Laflotte, A. Lath, R. Montalvo, K. Nash, H. Routray, S. Salur, S. Schnetzer, S. Somalwar, R. Stone, S. A. Thayil, S. Thomas, J. Vora, H. Wang

University of Tennessee, Knoxville, TN, USA
H. Acharya, D. Ally, A. G. Delannoy, S. Fiorendi, S. Higginbotham, T. Holmes, A. R. Kanuganti, N. Karunarathna, L. Lee, E. Nibigira, S. Spanier

Texas A&M University, College Station, TX, USA
D. Aebi, M. Ahmad, O. Bouhali[94], R. Eusebi, J. Gilmore, T. Huang, T. Kamon[95], H. Kim, S. Luo, R. Mueller, D. Overton, D. Rathjens, A. Safonov

Texas Tech University, Lubbock, TX, USA
N. Akchurin, J. Damgov, V. Hegde, A. Hussain, Y. Kazhykarim, K. Lamichhane, S. W. Lee, A. Mankel, T. Peltola, I. Volobouev, A. Whitbeck

Vanderbilt University, Nashville, TN, USA
E. Appelt, Y. Chen, S. Greene, A. Gurrola, W. Johns, R. Kunnawalkam Elayavalli, A. Melo, F. Romeo, P. Sheldon, S. Tuo, J. Velkovska, J. Viinikainen

University of Virginia, Charlottesville, VA, USA
B. Cardwell, B. Cox, J. Hakala, R. Hirosky, A. Ledovskoy, C. Neu, C. E. Perez Lara

Wayne State University, Detroit, MI, USA
P. E. Karchin

University of Wisconsin, Madison, WI, USA
A. Aravind, S. Banerjee, K. Black, T. Bose, S. Dasu, I. De Bruyn, P. Everaerts, C. Galloni, H. He, M. Herndon, A. Herve, C. K. Koraka, A. Lanaro, R. Loveless, J. Madhusudanan Sreekala, A. Mallampalli, A. Mohammadi, S. Mondal, G. Parida, L. Pétré, D. Pinna, A. Savin, V. Shang, V. Sharma, W. H. Smith, D. Teague, H. F. Tsoi, W. Vetens, A. Warden

Authors affiliated with an institute or an international laboratory covered by a cooperation agreement with CERN, Geneva, Switzerland

S. Afanasiev, V. Andreev, Yu. Andreev, T. Aushev, M. Azarkin, A. Babaev, A. Belyaev, V. Blinov[96], E. Boos, V. Borshch, D. Budkouski, M. Chadeeva[96], V. Chekhovsky, R. Chistov[96], A. Demiyanov, A. Dermenev, T. Dimova[96], D. Druzhkin[97], M. Dubinin[86], L. Dudko, A. Ershov, G. Gavrilov, V. Gavrilov, S. Gninenko, V. Golovtcov, N. Golubev, I. Golutvin, I. Gorbunov, A. Gribushin, Y. Ivanov, V. Kachanov, V. Karjavine, A. Karneyeu, V. Kim[96], M. Kirakosyan, D. Kirpichnikov, M. Kirsanov, V. Klyukhin, O. Kodolova[98], V. Korenkov, A. Kozyrev[96], N. Krasnikov, A. Lanev, P. Levchenko[99], N. Lychkovskaya, V. Makarenko, A. Malakhov, V. Matveev[96], V. Murzin, A. Nikitenko[100,98], S. Obraztsov, V. Oreshkin, V. Palichik, V. Perelygin, S. Petrushanko, S. Polikarpov[96], V. Popov, O. Radchenko[96], M. Savina, V. Savrin, V. Shalaev, S. Shmatov, S. Shulha, Y. Skovpen[96], S. Slabospitskii, V. Smirnov, A. Snigirev, D. Sosnov, V. Sulimov, E. Tcherniaev, A. Terkulov, O. Teryaev, I. Tlisova, A. Toropin, L. Uvarov, A. Uzunian, A. Vorobyev[†], N. Voytishin, B. S. Yuldashev[101], A. Zarubin, I. Zhizhin, A. Zhokin

[†] Deceased

1: Also at Yerevan State University, Yerevan, Armenia
2: Also at TU Wien, Vienna, Austria
3: Also at Institute of Basic and Applied Sciences, Faculty of Engineering, Arab Academy for Science, Technology and Maritime Transport, Alexandria, Egypt
4: Also at Ghent University, Ghent, Belgium
5: Also at Universidade Estadual de Campinas, Campinas, Brazil
6: Also at Federal University of Rio Grande do Sul, Porto Alegre, Brazil
7: Also at UFMS, Nova Andradina, Brazil
8: Also at Nanjing Normal University, Nanjing, China
9: Now at The University of Iowa, Iowa, USA
10: Also at University of Chinese Academy of Sciences, Beijing, China
11: Also at China Center of Advanced Science and Technology, Beijing, China
12: Also at University of Chinese Academy of Sciences, Beijing, China
13: Also at China Spallation Neutron Source, Guangdong, China
14: Now at Henan Normal University, Xinxiang, China
15: Also at Université Libre de Bruxelles, Bruxelles, Belgium
16: Also at an institute or an international laboratory covered by a cooperation agreement with CERN, Geneva, Switzerland
17: Also at Helwan University, Cairo, Egypt
18: Now at Zewail City of Science and Technology, Zewail, Egypt
19: Also at British University in Egypt, Cairo, Egypt
20: Now at Ain Shams University, Cairo, Egypt
21: Also at Purdue University, West Lafayette, IN, USA
22: Also at Université de Haute Alsace, Mulhouse, France
23: Also at Department of Physics, Tsinghua University, Beijing, China
24: Also at Ilia State University, Tbilisi, Georgia
25: Also at The University of the State of Amazonas, Manaus, Brazil
26: Also at Erzincan Binali Yildirim University, Erzincan, Turkey
27: Also at University of Hamburg, Hamburg, Germany
28: Also at RWTH Aachen University, III. Physikalisches Institut A, Aachen, Germany
29: Also at Isfahan University of Technology, Isfahan, Iran
30: Also at Bergische University Wuppertal (BUW), Wuppertal, Germany
31: Also at Brandenburg University of Technology, Cottbus, Germany
32: Also at Forschungszentrum Jülich, Juelich, Germany
33: Also at CERN, European Organization for Nuclear Research, Geneva, Switzerland
34: Also at Institute of Physics, University of Debrecen, Debrecen, Hungary
35: Also at Institute of Nuclear Research ATOMKI, Debrecen, Hungary
36: Now at Universitatea Babes-Bolyai - Facultatea de Fizica, Cluj-Napoca, Romania

37: Also at Physics Department, Faculty of Science, Assiut University, Assiut, Egypt
38: Also at HUN-REN Wigner Research Centre for Physics, Budapest, Hungary
39: Also at Punjab Agricultural University, Ludhiana, India
40: Also at University of Visva-Bharati, Santiniketan, India
41: Also at Indian Institute of Science (IISc), Bangalore, India
42: Also at Birla Institute of Technology, Mesra, Mesra, India
43: Also at IIT Bhubaneswar, Bhubaneswar, India
44: Also at Institute of Physics, Bhubaneswar, India
45: Also at University of Hyderabad, Hyderabad, India
46: Also at Deutsches Elektronen-Synchrotron, Hamburg, Germany
47: Also at Department of Physics, Isfahan University of Technology, Isfahan, Iran
48: Also at Sharif University of Technology, Tehran, Iran
49: Also at Department of Physics, University of Science and Technology of Mazandaran, Behshahr, Iran
50: Also at Italian National Agency for New Technologies, Energy and Sustainable Economic Development, Bologna, Italy
51: Also at Centro Siciliano di Fisica Nucleare e di Struttura Della Materia, Catania, Italy
52: Also at Università degli Studi Guglielmo Marconi, Roma, Italy
53: Also at Scuola Superiore Meridionale, Università di Napoli 'Federico II', Napoli, Italy
54: Also at Fermi National Accelerator Laboratory, Batavia, IL, USA
55: Also at Consiglio Nazionale delle Ricerche, Istituto Officina dei Materiali, Perugia, Italy
56: Also at Riga Technical University, Riga, Latvia
57: Also at Department of Applied Physics, Faculty of Science and Technology, Universiti Kebangsaan Malaysia, Bangi, Malaysia
58: Also at Consejo Nacional de Ciencia y Tecnología, Mexico City, Mexico
59: Also at Trincomalee Campus, Eastern University, Nilaveli, Sri Lanka
60: Also at Saegis Campus, Nugegoda, Sri Lanka
61: Also at National and Kapodistrian University of Athens, Athens, Greece
62: Also at Ecole Polytechnique Fédérale Lausanne, Lausanne, Switzerland
63: Also at Universität Zürich, Zurich, Switzerland
64: Also at Stefan Meyer Institute for Subatomic Physics, Vienna, Austria
65: Also at Laboratoire d'Annecy-le-Vieux de Physique des Particules, IN2P3-CNRS, Annecy-le-Vieux, France
66: Also at Near East University, Research Center of Experimental Health Science, Mersin, Turkey
67: Also at Konya Technical University, Konya, Turkey
68: Also at Izmir Bakircay University, Izmir, Turkey
69: Also at Adiyaman University, Adiyaman, Turkey
70: Also at Bozok Universitetesi Rektörlügü, Yozgat, Turkey
71: Also at Marmara University, Istanbul, Turkey
72: Also at Milli Savunma University, Istanbul, Turkey
73: Also at Kafkas University, Kars, Turkey
74: Now at stanbul Okan University, Istanbul, Turkey
75: Also at Hacettepe University, Ankara, Turkey
76: Also at Istanbul University, Cerrahpasa, Faculty of Engineering, Istanbul, Turkey
77: Also at Yildiz Technical University, Istanbul, Turkey
78: Also at Vrije Universiteit Brussel, Brussel, Belgium
79: Also at School of Physics and Astronomy, University of Southampton, Southampton, UK
80: Also at University of Bristol, Bristol, UK
81: Also at IPPP Durham University, Durham, UK
82: Also at Monash University, Faculty of Science, Clayton, Australia
83: Also at Università di Torino, Torino, Italy
84: Also at Bethel University, St. Paul, MN, USA
85: Also at Karamanoğlu Mehmetbey University, Karaman, Turkey
86: Also at California Institute of Technology, Pasadena, CA, USA
87: Also at United States Naval Academy, Annapolis, MD, USA
88: Also at Bingol University, Bingol, Turkey

89: Also at Georgian Technical University, Tbilisi, Georgia
90: Also at Sinop University, Sinop, Turkey
91: Also at Erciyes University, Kayseri, Turkey
92: Also at Horia Hulubei National Institute of Physics and Nuclear Engineering (IFIN-HH), Bucharest, Romania
93: Now at an institute or an international laboratory covered by a cooperation agreement with CERN, Geneva, Switzerland
94: Also at Texas A&M University at Qatar, Doha, Qatar
95: Also at Kyungpook National University, Daegu, Korea
96: Also at another institute or international laboratory covered by a cooperation agreement with CERN, Geneva, Switzerland
97: Also at Universiteit Antwerpen, Antwerpen, Belgium
98: Also at Yerevan Physics Institute, Yerevan, Armenia
99: Also at Northeastern University, Boston, MA, USA
100: Also at Imperial College, London, UK
101: Also at Institute of Nuclear Physics of the Uzbekistan Academy of Sciences, Tashkent, Uzbekistan