RESEARCH ARTICLE | DECEMBER 17 2024

Modeling performance of data collection systems for high-energy physics

Wilkie Olin-Ammentorp ; Xingfu Wu ; Andrew A. Chien

APL Mach. Learn. 2, 046113 (2024)
https://doi.org/10.1063/5.0232456

Modeling performance of data collection systems for high-energy physics

Cite as: APL Mach. Learn. 2, 046113 (2024); doi: 10.1063/5.0232456
Submitted: 7 August 2024 • Accepted: 1 December 2024 •
Published Online: 17 December 2024

Wilkie Olin-Ammentorp,[1,a)] Xingfu Wu,[1] and Andrew A. Chien[1,2]

AFFILIATIONS

[1] Argonne National Laboratory, 9500 S. Cass Ave., Lemont, Illinois 60439, USA
[2] University of Chicago, 5801 S. Ellis Ave., Chicago, Illinois 60637, USA

[a)] **Author to whom correspondence should be addressed:** wolinammentorp@anl.gov

ABSTRACT

Exponential increases in scientific experimental data are outpacing silicon technology progress, necessitating heterogeneous computing systems—particularly those utilizing machine learning (ML)—to meet future scientific computing demands. The growing importance and complexity of heterogeneous computing systems require systematic modeling to understand and predict the effective roles for ML. We present a model that addresses this need by framing the key aspects of data collection pipelines and constraints and combining them with the important vectors of technology that shape alternatives, computing metrics that allow complex alternatives to be compared. For instance, a data collection pipeline may be characterized by parameters such as sensor sampling rates and the overall relevancy of retrieved samples. Alternatives to this pipeline are enabled by development vectors including ML, parallelization, advancing CMOS, and neuromorphic computing. By calculating metrics for each alternative such as overall F1 score, power, hardware cost, and energy expended per relevant sample, our model allows alternative data collection systems to be rigorously compared. We apply this model to the Compact Muon Solenoid experiment and its planned high luminosity-large hadron collider upgrade, evaluating novel technologies for the data acquisition system (DAQ), including ML-based filtering and parallelized software. The results demonstrate that improvements to early DAQ stages significantly reduce resources required later, with a power reduction of 60% and increased relevant data retrieval per unit power (from 0.065 to 0.31 samples/kJ). However, we predict that further advances will be required in order to meet overall power and cost constraints for the DAQ.

I. INTRODUCTION

Progress in many scientific disciplines depends on the ability to collect and analyze increasingly large volumes of experimental data. Scientific experiments including x-ray microscopy, radio astronomy, characterization of neutrinos, and exploration of high-energy physics (HEP) will produce more data than in previous decades.[1–4] HEP in particular is data intensive: detectors currently produce over 80 TB/s, with future increases planned.[5] This growth is a natural result of progress: refinement of extant knowledge requires higher precision, and discovery of new phenomena requires searching further and in greater detail than previously possible; achieving either goal in such experiments depends on an increased ability to carry out experiments and process data.

Historically, increased demand for data processing in scientific experiments was accompanied by multiple improvements in computing technologies, particularly in the areas of processing and communication. Processors became faster, memory became less expensive, and communication links improved in capacity and efficiency.[6] These advances enabled processing and transmitting greater amounts of data without a corresponding increase in power and space requirements. However, shifts in trends of microelectronic development—such as the end of Dennard scaling and the slowdown in Moore scaling—have reduced the rate of improvement in computing.[7]

This situation has motivated new trends in computer software and architecture. One high-profile advancement has been the growing capability of machine learning (ML) techniques, which has motivated a growing variety of ML-focused components to be brought to market.[8] In general, computing components focused on improving compute performance within certain domains are becoming commonplace and include graphics processors, network processors, and neuromorphic systems.[9] The broadening availability

of these components provides new opportunities for the designers of computing systems, but evaluating their application—where and how they should be integrated into computing systems to provide advantages over conventional approaches—remains a novel challenge. ML-based approaches must ultimately be effective not only in isolated tests and validation, but provide net gains in efficiency and effectiveness when integrated into larger computational systems.

We provide an analytical, predictive model to evaluate the opportunities offered by novel approaches for computing systems. This "SystemFlow" model utilizes properties from individual hardware components to estimate the overall flow of information through a real-time computing system. Identifying these properties allows researchers from multiple disciplines, from devices to algorithms, to examine the impact their contributions can make toward improving system-level properties such as total power, effectiveness at retrieving relevant experimental data, number of computing devices required, and vectors toward future system improvement. This empowers researchers to systematically evaluate the role of emerging devices and architectures within large-scale computing systems.

We demonstrate the application of this model to examine the performance of HEP data acquisition systems at the Compact Muon Solenoid (CMS) experiment at the Large Hadron Collider (LHC),[10] and we quantify the benefits of multiple component-level changes. These include introducing novel ML-based data reduction processors, improving the skill of classifiers, integrating parallelized algorithms, and varying system-level parameters such as data retention rates.

II. BACKGROUND

A. Computing trends

The end of Dennard scaling in the mid-2000s and the slowdown in the scaling of transistors and other devices have motivated numerous shifts in computing. In software, parallelism of computer programs has become highly important, and ML capabilities have developed into a powerful tool that can introduce entirely new capabilities into processing systems and approximate expensive computations using fewer resources.[11–13] In hardware, commercial systems have begun to more broadly leverage specialized instructions, representations, and circuit topologies.[9]

As a result, modern high-performance computing systems are constructed not only from CPUs but also from a variety of components, including graphics processing units (GPUs), AI accelerators, field-programmable gate arrays (FPGAs), and application-specific integrated circuits (ASICs).[8,14,15] However, each of these components remains fundamentally based on the same technology: digital electronics. Therefore, to project each of these hardware components' future capabilities, we assume that they will correspond to the gradually slowing improvements in digital electronics. In particular, we assume that circuit density will continue to increase while maintaining an approximately constant per-element operating speed and total power dissipation level. For instance, CPUs, GPUs, and accelerators will provide more computing throughput (operations, "ops") at approximately constant speeds and total power.

In concert with processing, communication technologies play a crucial role in enabling computational systems. Edholm's law of bandwidth describes the scaling of wired and wireless communication that, similarly to Moore's law, has demonstrated exponential growth in the capacity of communication channels.[16] Fast and efficient communication is needed for computing systems across multiple length scales, as demand for bandwidths between heterogeneous computing modules and memories has continued to increase, particularly in ML where computations are often bound by accessing data from memory. Developments in packaging, photonic circuits, and more continue to enable higher bandwidths as well as novel trade-offs between cost, capacity, efficiency, and transmission distance.[17–19]

The design and integration of novel processing and communication technologies into a computing system require significant engineering effort. To evaluate the benefits and trade-offs offered by novel technologies, one must measure their ultimate impact on system performance in comparison with employing standard technologies. HEP is one data-driven discipline that has applied computer-based analysis and experimental control for decades. To demonstrate the impacts that novel technologies may have on future experiments, we utilize HEP data acquisition systems (DAQs) as a case study.

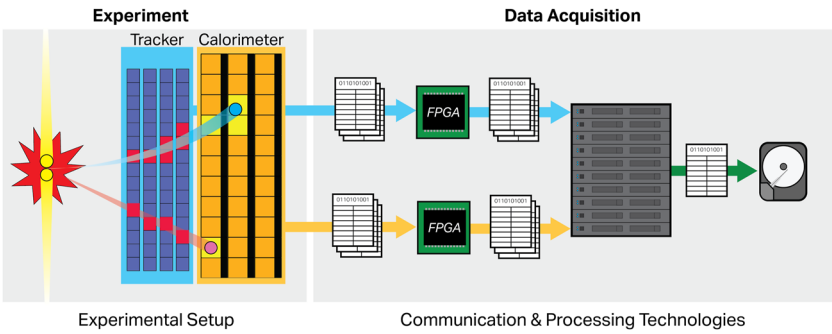


FIG. 1. The amount of data that an experiment produces must be matched with a downstream computational system that meets or exceeds its needs. In high-energy particle colliders, experimental variables such as luminosity and the resolution of detector systems influence the overall amount of data produced. Systems that analyze and classify the data require communication and processing systems. The power, area, and scalability of these systems are influenced by the specific technology being deployed.

B. Data and computing in high-energy physics

The identification of fundamental physical particles and interaction mechanisms is a long-standing scientific goal, addressed over the previous century by refining theoretical models and collecting evidence to support or refute these theories. In order to accomplish this, bunches of particles are accelerated and directed to collide at precise locations surrounded by detectors. The products of each collision are measured through properties such as electromagnetic traces and energy deposits.[10,20] These measurements allow for the identification of individual particles, their energies, and trajectories. Determining these parameters with high confidence enables a post hoc analysis to "reconstruct" the full interaction that led to a given set of products, identifying intermediate stages that are not directly measured by the detector (Fig. 1).[21]

The Compact Muon Solenoid experiment is one of two general-purpose detectors installed at the LHC. Under current conditions, the detector records ~2.0 MB of data each time two bunches of particles cross at the interaction point.[5] These samples, termed "events" in HEP, are produced at the bunch crossing rate of 40 MHz, producing a combined output rate of ~80 TB/s—a volume of data that cannot sustainably be stored over months of experimental runs.

However, counterbalancing this torrent of data is the scarcity of results that are believed to contain novel data relevant to current physics experiments. For instance, models predict that currently the LHC produces a Higgs boson only once every few seconds.[22] In order to maintain a sustainable output, the CMS detector's current (phase-1) DAQ is configured to ultimately save only 1 out of every 40 000 events, selecting for later analysis only those with specific signatures.[4,23] Currently, most DAQ systems utilize multiple processing stages to carry out this selection process in real time; to select events of interest, the DAQ must achieve a sensitivity on the order of parts per billion (Fig. 1).

The objective of the CMS DAQ is to maximize the capture of novel information for physics experiments, such as characterizing the Higgs boson and searching for evidence of supersymmetric particles. These data may be rare; with samples being produced at a rate of 40 MHz, this implies that the selectivity of the DAQ must exceed parts per billion. This leads to strict requirements for algorithms that can detect relevant features from sensor data with high confidence and speed. The CMS DAQ utilizes two major processing stages to achieve this high selectivity: the Level-1 Trigger (L1T) and High-Level Trigger (HLT).

In its current phase-1 DAQ configuration, the L1T utilizes information from the CMS muon and calorimetry systems to infer the presence of particles with high momenta (such as muons, tau leptons, electrons, and photons), as well as groups of particles with high energy (such as hadronic jets). These markers indicate that a rare, high-energy interaction may have taken place. Algorithms to examine detector data for these features are implemented in FPGA hardware located near the detector but in a radiation-safe zone. These results are used to down select samples produced by the detector, with algorithms calibrated to pass, on average, one in every 400 samples to the next stage of analysis.

The HLT utilizes more complex algorithms and all available data from CMS detector systems to "reconstruct" the event that led to the patterns recorded by the detector. This is a much more energy-intensive process than the simple feature detection used in the L1T; in the phase-1 HLT, software implementing the reconstruction algorithms is run on commodity hardware (CPUs) in a conventional datacenter. This process further down selects samples by a ratio of 1 to 100, leading to a final sample output rate of 1 kHz.

Numerous strategies for advancing the capabilities of DAQ systems leverage growing ML capabilities for tasks such as momentum prediction, track reconstruction, jet flavor tagging, and anomaly detection.[24–26] These capabilities may be applied to reduce the latency of an analysis and improve its throughput and performance while reducing the amount of energy and material it requires to execute. In order to evaluate the impact novel hardware and algorithms may have on performance of DAQ and other real-time computing systems, a systematic evaluation model is needed. Applying our performance model, we perform a case study on the CMS DAQ to estimate the impact of novel technologies, including an ML-based data reduction method.

III. THE SYSTEMFLOW MODEL

A. Overview

Most modern scientific experiments produce digital data that must be analyzed and stored by a computing system. In order to avoid losing valuable data, a computing system must be matched to meet the requirements of an experiment. We utilize several key descriptors to capture both the needs of a scientific experiment and the attributes of technologies that may be deployed to meet those needs. These descriptors are shown in Fig. 2.

B. Experimental characterization

A scientific experiment that produces digital data has several basic properties that predict the basic needs a computing system must meet. We predict these needs through the metrics of sensor occupancy (the proportion of sensing elements producing a non-zero result), the maximum number of experimental samples produced per second (Hz), and the proportion of samples that are relevant to scientific goals (%).

C. Technological characterization

A computing system's components may be partitioned into four major categories: input, communication, processing, and output. Inputs collect information from the environment. Communication technologies enable the transport of information between different locations. Processing technologies utilize algorithms to transform information. Outputs may store information or present it to the environment.

The arrangement of these components into a computing system may be represented as a directed graph, where inputs, outputs, and processing are represented by nodes and are connected via communication links (edges). In the case of data analysis systems for experiments, these graphs may take the form of trees, where information flows "upward" from input sensors to intermediate processing stages, eventually coalescing at the system's outputs. The requirements and performance of the system depend on the interaction of each part as information flows up from the sensors toward storage, implementing a processing pipeline.

We model the propagation of information through such a processing pipeline. This is accomplished by identifying the amount of information produced by each node and propagating it up the tree.

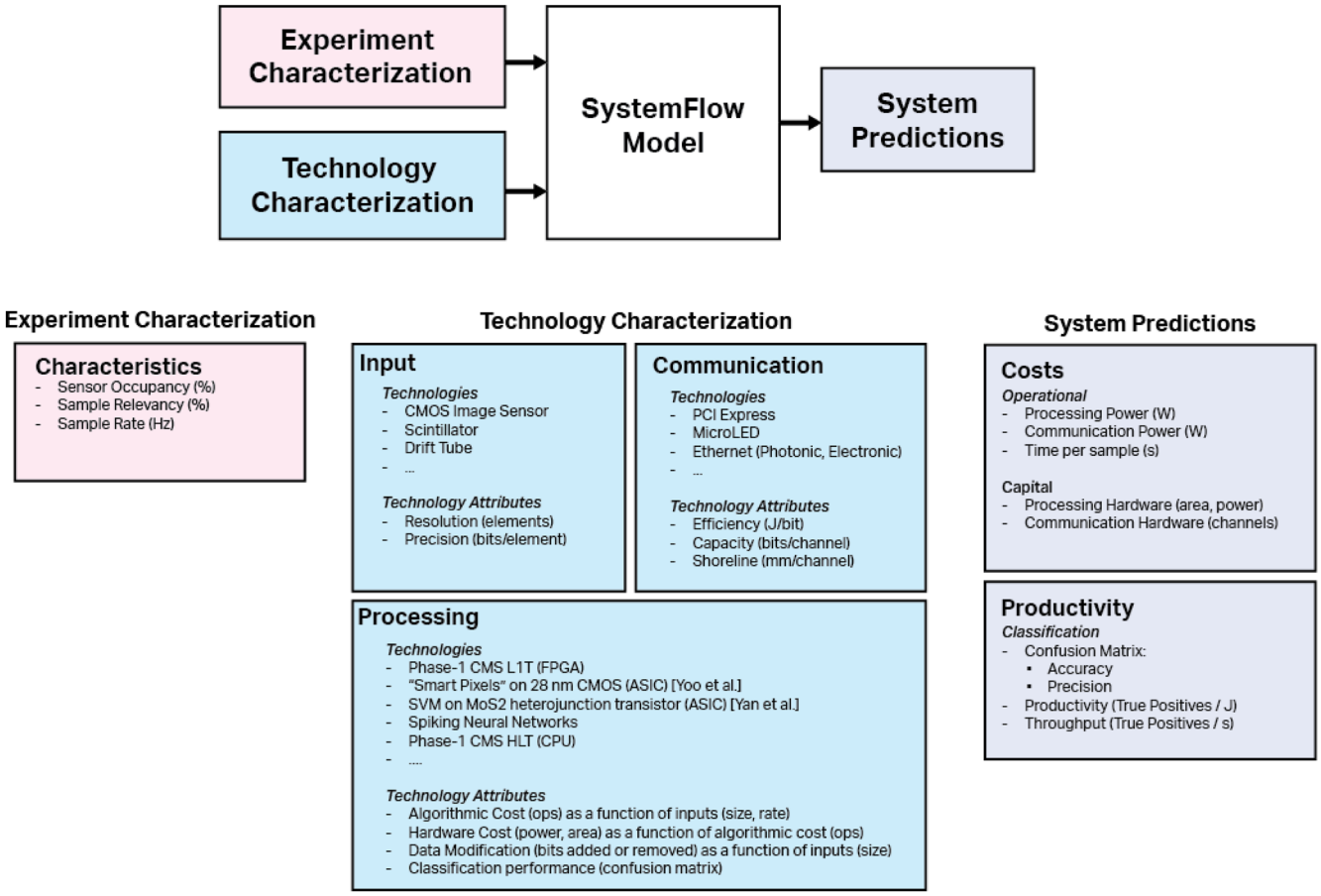


FIG. 2. In a SystemFlow model, components of a computing system are classified into inputs, processing, and communication. Each category utilizes metrics and functions to capture how it transforms information and the requirements needed to do so. Each alternative technology for sensing, communication, and processing will have a unique set of attributes. By simulating the flow of messages containing information through this system, the attributes of each component interact to estimate component and system-level outputs, costs, and productivity metrics (such as processing power, communication channels, and number of true positive samples produced per second).

Information is contained in discrete "messages" that originate at input nodes with a given frequency. Communication links transport these messages to processing stages. These stages implement algorithms that can change the amount of information in the message, increasing it by providing new analyses or reducing it by applying compression or removing irrelevant data. In addition, information from analysis may be used to make a decision such as to discard a message. The propagation and modification of messages are simulated until each one either is discarded or reaches an output node. Since this model describes the flow of messages through a computing system, we term this approach the "SystemFlow" model.

1. Inputs

We define an input as a node that records information from the environment with a set number of elements (resolution) and precision. Examples include a camera recording an image from millions of pixels with 8-bit color or a spectrometer recording 1024 frequencies with 16-bit precision. The constants defined by the experimental characterization and input design determine the size of messages (bits), proportion of messages relevant to the experiment (%), and the rate at which they are introduced into the system (Hz). These parameters are determined empirically (experimental environment) or by the system designer (sensor design).

2. Communication

Communication links are responsible for transporting information between different logical units in a computing system. These may take many different forms, from buses implemented via metal traces in a silicon wafer to long-range photonic links. Each is characterized by several parameters, chiefly latency (s), bandwidth [bits/(second · channel)], efficiency (J/bit), and area requirements or "shoreline" (mm/channel). These are determined empirically for each available technology.

3. Processing

Processors implement algorithms chosen by system designers to carry out analyses relevant to a given message. Examples include classical algorithms such as binning, peak detection, and graph

analysis, as well as ML techniques for object recognition and classification. In hardware, each algorithm is realized by discrete processing steps that are taken for it to execute (ops). The number of ops may vary with the size of an incoming message and the algorithm. In the best case, ops remain constant with the size of an input; more often, they will scale linearly, polynomially, or even more aggressively. Furthermore, each algorithm may require greater or fewer ops for messages of identical size based on its contents (i.e., more "complex" data incur more branching and processing steps). We avoid the latter complexity by assuming that the number of messages passing through the system is large, and an average processing time per message of a given size suffices to estimate a functional relationship between the two metrics (bits and ops). Empirically or theoretically capturing the functional complexity for an algorithm on a processing technology translates informational requirements into hardware requirements; capturing this relationship is necessary to estimate how changing experimental conditions will induce new requirements for processing hardware. Knock-on requirements for each op, such as silicon area, power, and cost, can then be estimated for each hardware system. Functional scaling of ops and knock-on requirements varies for different technologies (e.g., CPU, GPU, and ASIC) and may be modeled empirically and/or theoretically.

After executing the necessary algorithm, a processing node will pass a message containing a result to the next node up the tree via a communication link. The size of this message will vary based on the processing stage; analyses may add information (such as high-level objects recognized) or remove information (transforming a full vector of wavelengths into a small list of peaks). This is measured in bits and may remain constant or vary with the size of the input.

One subset of algorithms induce an additional consideration: classification algorithms can determine whether an incoming message is relevant to an experiment. When irrelevant, a message is dropped at the given processing node and does not propagate up the tree toward the output. In order to capture the performance of this classification process, the distribution of scores produced by the classifier analyzing relevant and irrelevant populations is modeled. Capturing this relationship is necessary to estimate how modifying a classifier to vary its sensitivity threshold will impact the performance of the overall system. These score distributions may be estimated empirically or parametrically and produce a confusion matrix for each selected threshold.

4. Output

The messages that propagate up the processing tree are captured as the system's output. This supplies the rates that must be sustained by the storage system (bits/s), as well as the number of relevant and irrelevant messages (true and false positives) that have propagated up the processing tree. These statistics may be used to estimate the overall productivity of the pipeline by calculating accuracy, precision, and the statistics normalized by the overall cost (operational and capital expenditure). In contrast to the other system components, output characteristics are not defined as a part of the system's definition but estimated as a response to it.

D. Model outputs

To summarize, the propagation of messages through the system models the knock-on effects of multiple processing stages, each with unique costs and performance characteristics. These properties may be investigated in the model at each stage or across the system as a whole. For instance, the power of a single processing stage or the number of channels in one communication link may be inspected, as well as the power of the system as a whole or its overall performance at separating significant and insignificant samples.

Furthermore, these estimates scale to different system-level design parameters, such as the technology utilized at each stage and varying experimental conditions. This capability enables system architects to not only model a system in its current configuration but also estimate how its requirements and performance will vary as system technologies or experimental conditions are altered.

IV. CASE STUDY: THE CMS DAQ

A. Current (phase-1) DAQ

The phase-1 configuration (described in Sec. II B) is utilized as the baseline against which alternative systems are compared.

B. Future HL-LHC experiments

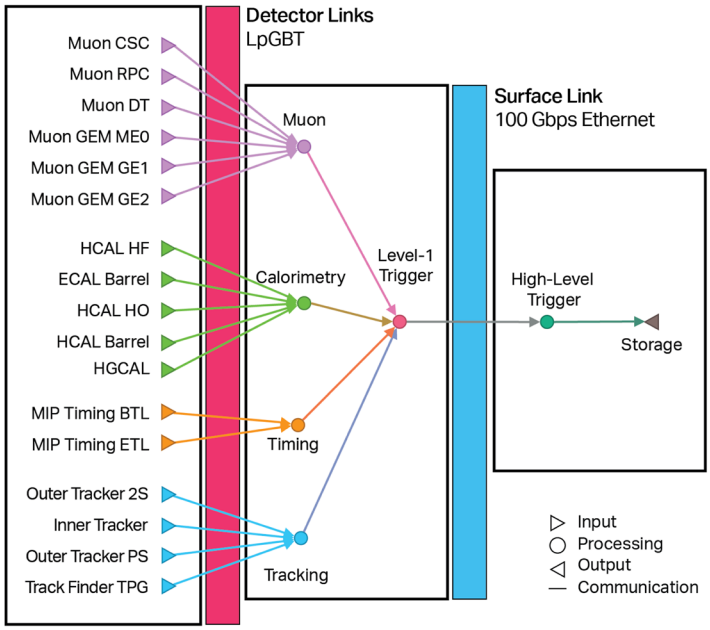
Currently, the LHC is undergoing its final experimental run (run-3) before upgrades are made to its accelerator systems. The objective of these upgrades is to increase the precision with which bunches are collided in the detector, leading to more high-energy collisions per bunch crossing. This will increase the overall luminosity of the LHC, leading to its "high-luminosity" (HL-LHC) configuration. In subsequent HL-LHC runs in 2028 (run-4) and 2032 (run-5), the stated goal is to achieve pileups of 140 and 200, respectively.[5]

Under the HL-LHC configuration, the CMS detector will record more data per sample: an increase in pileup leads to more particles yielded from each collision, causing the tracker to register more charged tracks and the calorimeter to record more energy deposits. This will lead to an increased size of 8.0 MB per sample and a total detector data rate of 320 TB/s during run-5.[5] Increasing the number of particles also leads to greater complexity in analysis. Simulations show that the HLT analysis of run-5 events will take ~16 times longer than the analysis of run-3 events with current algorithms and hardware.[5] Moreover, the number of events passed from the L1T to HLT is planned to increase 7.5 times, increasing the L1T output rate from an average of 100 to 750 kHz. The motivation behind this change is to utilize the higher selectivity of the HLT to retain events that might otherwise be rejected by the L1T and to provide more experimental data.

These factors will increase the computational load that the DAQ must sustain. To quantify this increase, we adapt the current (phase-1) CMS DAQ into a SystemFlow model. The parameters of this model are then shifted to simulate the conditions under the final planned HL-LHC configuration, run-5.

C. SystemFlow DAQ model

The CMS DAQ is adapted into a SystemFlow model by utilizing a breakdown of CMS detector subsystems to identify all input nodes and the amount of data they produce. Processing nodes are created for each detector subsystem, the L1T, and HLT. The HLT provides the output of the system that is archived.[5,10,27,28] This SystemFlow model of the CMS DAQ is shown in Fig. 3.



	CMS Detector	Level-1 Trigger	High-Level Trigger
Location	Experiment Cavern	Service Cavern	Surface Data Center
Output Sample Rate	40 MHz	100 kHz	1 kHz
Output Data Rate	80 TB/s	200 GB/s	2 GB/s
Processing Latency	25 ns	3.8 μs	300 ms
Approx. Power	70 kW	120 kW	1600 kW
Hardware	ASIC	FPGA	CPU

FIG. 3. Illustration of the CMS DAQ system adapted into a SystemFlow model. Samples originate at sensors providing input to the computing system (left). These samples are transported via communication links to processing nodes (middle), where they may change in size and/or be discarded. These changes propagate downstream to further processing nodes and, eventually, the output (right). The metrics necessary to characterize each node and edge in the SystemFlow model are taken from public documentation.

1. Detectors

All detector subsystems within the CMS are included as input nodes in the SystemFlow model. The effect of pileup on amount of incoming data is included by scaling the bits recorded by each sensor by pileup according to CMS estimates.[5] A new detector system, the high-granularity calorimeter will be installed in the CMS after run-3 and incorporated into the detector for run 4 and run 5. The significant amounts of data produced by this subsystem are included in the estimates for both runs.

2. L1T

Data are transported from sensors to the L1T by a custom radiation-hardened link, the low-power gigabit transceiver. Public data place the efficiency of this link at 22 pJ/bit.[28] Algorithms used within the L1T identify features such as high-energy particles and "jets."[4] These relatively simple algorithms search for clusters and peaks, and we estimate that the resources required for these algorithms will scale approximately linearly with the amount of incoming data from sensors. The performance of these algorithms at classifying the incoming samples as "relevant" or "irrelevant" is modeled in detail on the performance of identifying each high-energy feature. For details, see Subsection 1 of the Appendix. The power efficiency of the L1T is scaled to its current needs, ~120 kW.[28] Currently, the L1T retrieves tracking data only after identifying a feature of interest from other data—this feedback behavior is not included in the current model, but it produces no significant difference in the metrics predicted by the SystemFlow model, impacting only communication costs from the detector to the trigger, which are less than 1% of overall system usage.

3. HLT

Data accepted by the L1T are transmitted to the HLT datacenter by standard 100 gigabit Ethernet over fiber; commercially available medium-link transceivers require ~25 pJ/bit.[29] Complex algorithms are utilized within the HLT to identify particle tracks, associate them with calorimeter hits, identify intermediate particles produced by the collision, and more. The scaling of these algorithms with increased incoming data is modeled empirically by collating data from CMS and ATLAS experiments estimating the runtime of HLT reconstructions under increasing pileup.[5,20] The power efficiency of the HLT is scaled to its current consumption, ~1.6 MW.[5] The HLT is much more complex than the L1T, with hundreds of separate "paths" allowing an event to be selected for offline storage. The overall performance of the HLT as a classifier is modeled by utilizing public results detailing the performance of six different trigger paths used for different experiments within the CMS (e.g., Higgs, supersymmetry, and exotica—see details in Subsection 4 of the Appendix).

V. RESULTS

A. Phase-1 system

First, we examine the characteristics of the current (phase-1) CMS DAQ as if it were scaled to meet the desired experimental goals

TABLE I. SystemFlow estimates of phase-1 DAQ productivity by pileup and L1T reduction ratio.

Pileup	L1T Reduction ratio	DAQ power (MW)	Precision (%)	Recall (%)	F1 score (%)	Productivity (1/kJ)
60	400:1	0.32	28	28	28	0.86
200	400:1	7.0	23	24	24	0.034
200	53:1	52	40	40	45	0.060

for run-5: a collider pileup of 200 and a decrease in the rejection ratio of the L1T from 400:1 to 53:1 (increasing the effective level-1 "trigger rate" from 100 to 750 kHz and overall output of the DAQ to 7.5 kHz). Hardware and algorithms within the system are held constant, but hardware during run-5 (2032) is modeled as being 6.5 times more energy-efficient than at present (2024) based on a corresponding increase in the areal density of integrated circuits operating at constant power (see Sec. II A). DAQ productivity is measured in the expected number of relevant samples produced per second normalized by the power taken to acquire them (1/J). This is produced from each SystemFlow model by multiplying the total output rate by its system-level F1 score (1/s), normalized by power used across the entire pipeline (J/s),

$$\text{productivity} = \frac{\text{output rate}}{\text{power}} \cdot \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}.$$

Meeting the run-5 experimental goals will have a large impact on the DAQ. Increasing pileup from 60 to 200 greatly lengthens the amount of runtime needed to process a sample within the HLT, due to the increased complexity of the underlying data. In particular, processes such as reconstructing particle tracks scale poorly with increasing pileup.[28,30] Furthermore, in order to retain increased amounts of relevant information, 7.5 times more samples will be sent from the L1T to the HLT. Combining these changes causes overall DAQ power requirements to increase significantly, from 0.32 to 52 MW, with the increase almost entirely stemming from CPU-based HLT processing (Table I). The productivity experiences a net drop from 0.80 to 0.065 1/kJ due to the greatly increased power requirements for a moderately increased rate of relevant samples retrieved by the system (Fig. 4).

Scaling the baseline DAQ system to the needs of run-5 is thus undesirable, representing approximately a 20 times increase in power expenditure compared with the phase-1 DAQ during run-3. This motivates the changes to the system whose impacts we investigate next.

B. Next-generation systems

We quantify and compare the impact of three major changes proposed to improve the CMS DAQ as it scales to satisfy run-5 conditions: incorporating tracking information into the L1T, utilizing GPU-based hardware in the HLT, and utilizing ML-based detector-integrated data filtering in the CMS Inner Tracker ("smart" sensing). The impact of these individual changes on the overall DAQ system is modeled by introducing corresponding changes into the SystemFlow model.

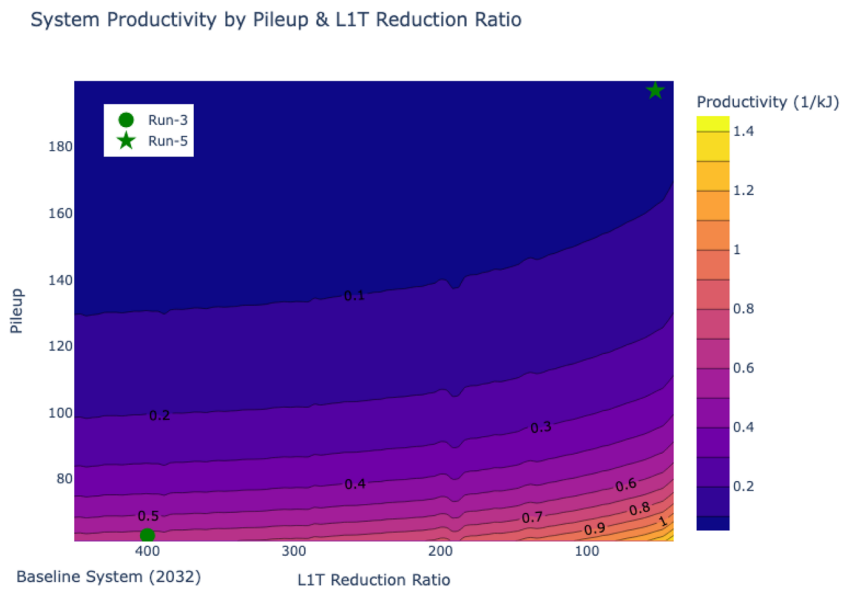


FIG. 4. Increasing the pileup of collisions within the detector and increasing the number of samples passed from the L1T to the HLT significantly changes the productivity of the overall DAQ system. As conditions shift from run-3 (bottom left) to run-5 (top right), net productivity first increases by reducing the L1T's rejection ratio. However, the energy cost of processing more samples within the HLT as pileup increases causes productivity to drop steeply.

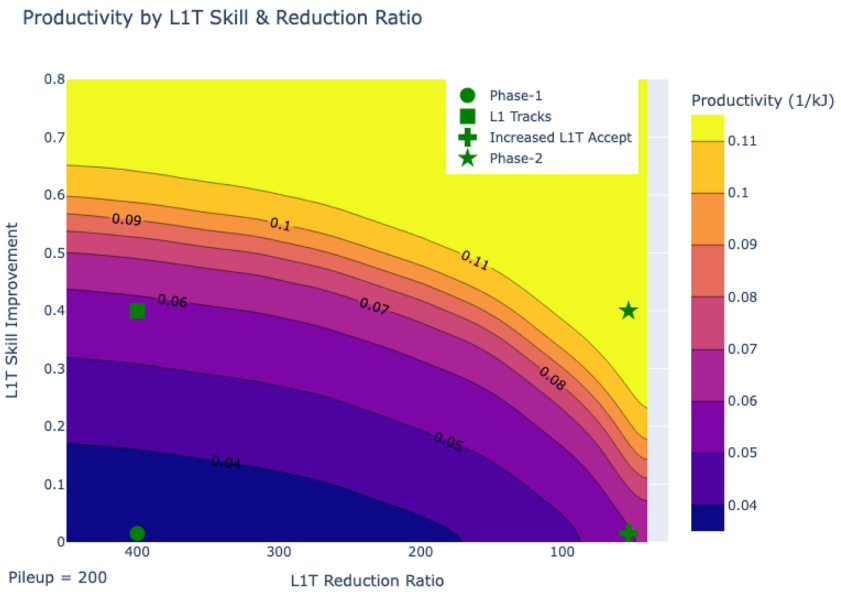


FIG. 5. Productivity of a DAQ system operating on an experiment with an average pileup of 200 is plotted as the reduction ratio of the L1T is reduced and its skill is increased. Introducing higher skill into the L1T is highly beneficial for the overall system productivity, yielding greater gains than simply passing more samples to the HLT.

1. L1 Tracks

Tracking information is not utilized in the phase-1 L1T, making certain analyses infeasible, such as distinguishing charged and uncharged particles (e.g., electrons and photons). Future upgrades to the L1T system focus on incorporating tracking information and making other algorithms available to improve the selectivity and capabilities of the L1T system.[28] We explore the impact that improved L1T classification skill has on the DAQ by increasing the separation between scores produced by positive and negative samples in a model of the phase-1 L1T classifier (see Subsection 1 of the Appendix for details).

Our model predicts that improving the performance of classification within the L1T can lead to significant improvements in overall system performance; improving the separation of scores between positive and negative samples within the L1T by 40% can lead to the same improvement in overall system classification performance as increasing the number of samples sent from the L1T to the HLT by 750% (Fig. 5). Since the L1T is responsible for discarding the majority of samples collected from the detector, any improvement to it significantly improves the entire DAQ.

2. GPU HLT

Many algorithms utilized within the event reconstruction carried out in the HLT can be effectively parallelized, allowing software to take advantage of hardware such as GPUs to increase throughput. Experiments using parallelized routines running on GPU hardware for CMS event analysis show an increase in the number of samples that can be processed per second by using a GPU (such as an approximate 50% increase using an NVIDIA A100).[5] This change is

TABLE II. System-level power consumption, classification performance, and overall productivity as various changes are implemented into the CMS DAQ. "GPU" corresponds to the utilization of GPU hardware to parallelize HLT algorithms for event selection. "L1 tracks" corresponds to the introduction of tracking information in the L1T, improving the performance of its classifier. "Smart pixels" corresponds to the introduction of smart sensing elements within the inner tracker that remove unnecessary data from samples.

System	DAQ power (MW)	Precision (%)	Recall (%)	Productivity (1/kJ)
Phase-1	52	40	40	0.059
GPU HLT	26	38	39	0.11
L1 tracks	52	100	79	0.10
Smart pixels	41	40	40	0.074
GPU and L1 (phase-2)	26	100	79	0.20
Smart GPU	41	100	79	0.13
Smart L1	20	38	39	0.14
Smart phase-2	20	100	79	0.26

captured in the SystemFlow model by adjusting the complexity function of the HLT processing node to match the throughput results of the parallelized algorithms. Simultaneously, the power for the node is modified to match the average dissipation of the NVIDIA SXM4 A100 accelerator (400 W). Incorporating this change into the DAQ system would roughly double the system productivity to 0.13 1/kJ, because HLT consumes the vast majority of processing energy (Table II). Aggressive improvements to GPU hardware could further improve the energy efficiency of traditional HLT algorithms; however, this approach may be limited by development trends in GPUs focusing on limited-precision numerical representations adapted to artificial intelligence (AI) system needs.[31]

3. Smart sensing

Processing adapted to specific sensors has been proposed as a method to reduce the amount of data that must be transported from detector systems while simplifying subsequent processing. For instance, high-resolution pixel sensors with on-chip neural networks can be used to detect and reject data from low-momentum particles irrelevant to later analysis.[24] The impact of this change on the DAQ is modeled after this result by assuming that neural network ASICs integrated into next-generation tracking modules (smart pixels) could reduce the amount of data from an inner tracker pixel module by 54% or more, dissipating 300 μW to analyze 256 pixels. We assume that this approach could scale effectively to the CMS Inner Tracker, which has 2×10^9 pixels and would require at least 2.3 kW to implement a classification algorithm co-located with detection elements.

Training ML methods such as neural networks can be a costly stage of development that can require large and specialized resources. However, given the small network used for the task—a multilayer perceptron with only 145 neurons and 2176 synapses—the training cost in this case is quite low. Even using nonoptimal training conditions, we estimate that the energy dissipated for training would be overtaken by inference within the detector after 160 s of operation in experimental runs that last 12 h or more. As a result, in this case, training the network requires less than 1% of the total energy used by the ML system, even where the network is retrained for calibration twice a day (see Subsection 1 of the Appendix for details).

We estimate that for the cost of dissipating several kilowatts of power in the detector, a "smart pixels" approach discarding low-momentum particles constituting 54% of tracking data could ultimately lead to a net reduction of 6 MW over the next-best DAQ system, greatly increasing the overall system performance (Table II). This saving is enabled by leveraging a simple algorithm adapted to specialized hardware utilizing local information to avoid more complex analysis later in the pipeline.

C. Discussion

1. Prospects for future DAQ systems

Through our case study, we have demonstrated that depending solely on Moore's law scaling to meet the future demands of HEP experiments would require greatly increased capital and operational expenditures if changes to the processing system (DAQ) are not implemented. As the cost of computing becomes a larger issue in society, simply accepting these increased expenditures is not an acceptable solution to meet these future demands.

In anticipation of this challenge, several approaches are being developed. We modeled the impact of improving the DAQ's L1T, utilizing GPU-based processing in the HLT datacenter, and implementing ML-based "smart" detector elements in the front-end of the system. Each of these changes carries unique consequences; improvements to the L1T greatly improve the quality of results sent to the next stage of processing, the HLT, sending fewer false positives and discarding fewer false negatives with minimal impacts to system power. The HLT's efficiency can be improved through the use of parallelized computation carried out on GPUs, roughly halving the amount of power utilized by the DAQ. The net system power can be further reduced by eliminating unneeded data from samples by using ML-based "smart" sensing strategies.

Overall, including these three changes greatly boosts the overall productivity of the DAQ to 0.31 1/kJ (Table II); this exceeds our estimate of the current productivity of the phase-1 HLT, 0.13 1/kJ. However, given the objective of the DAQ not only to process more complex samples created under the conditions of increased pileup but also to produce more samples of interest to experimenters, this productivity must be further improved. To reach the goal of producing a relevant sample rate of 7.5 kHz without significant increases to power, the DAQ must reach 0.98 1/kJ. Achieving this three times improvement may require more aggressive changes to the DAQ, such as adapting approximate or limited-precision computations as well as utilizing ML-assisted triggers.

2. Guiding future ML research

ML has been demonstrated as a tool that can improve existing algorithms or enable entirely new approaches toward problem-solving: dozens of application areas for ML in HEP have been demonstrated. The potential incorporation of each ML strategy into a processing pipeline represents a large space of discrete changes that would be difficult to comprehensively evaluate for the most promising directions.[32] However, individual SystemFlow models contain valuable information that can be used to find development vectors that can be targeted by ML systems and predict the outcome of design decisions.

For instance, ML techniques may potentially be applied to improve the skill of HLT classifiers via improved jet tagging, or potentially reducing the cost of HLT classification by replacing many individual triggering paths via a unified anomaly detection network.[33,34] In the context of a SystemFlow model, the former approach could potentially increase the effectiveness of the HLT classifier, while the latter could decrease its energetic cost. To identify the relative merits of each trade-off, the gradient between the model's productivity and these input parameters is measured. Gradients between system parameters and overall productivity from the final evaluated system configuration (smart phase-2, Table II) estimate that reducing the energetic cost of HLT evaluation by a Joule would be approximately twice as effective in raising system productivity as improving the skill of the HLT classifier by an entire point. This simple metric quantitatively highlights the relatively high performance and cost of the extant HLT and suggests that effective approaches in ML for HEP should more aggressively target this energetic cost rather than improving classification performance.

Rather than applying ML-based jet tagging to the HLT, we suggest that further improvement of the level-1 trigger's skill via ML-augmented methods could provide greater utility to a DAQ system. Model gradient estimates show that improving one point in classifier skill of the L1T vs HLT contributes 75% more to overall system productivity, although this must be accomplished under a constant or smaller-than-present energy envelope. However, this should be feasible given that even very small neural networks can identify advanced features.[35]

To summarize, gradient information present in individual SystemFlow models can be used to quantitatively predict the relative merits of targets for ML techniques: we estimate that methods for utilizing anomaly detection to lessen the cost of HLT evaluation would be more effective than those aimed at improving the HLT's skill via jet-tagging and that instead very small and efficient versions of these techniques would be more beneficial when applied to the L1T.

3. Automated design

The gradients provided by SystemFlow models could potentially be combined with optimization systems to discover what appear to be optimal design points. That is, given the number of system-level parameters or "dials" that can be turned on by the model, optimization tools could be used to automatically discover system configurations that maximize metrics such as productivity. While such an approach is appealing, we argue that this is not an appropriate use of the tool on several grounds.

First, we have applied SystemFlow to explore a set of discrete design choices that belong to a larger, innumerable space. We introduced new changes to the CMS DAQ including GPU implementations and neural-network-based data filtering, but many other system-level changes are under consideration and could be introduced. Therefore, the ability to tune the "dials" of a single system to what appears optimal does not allow for the larger design space to be considered. To make an analogy to neural networks, we may "train" a system to reach a minimum on a metric, but the larger—and often more important—process of architecture search is not addressed.

Second, each design choice carries with it a number of assumptions that may be valid at one operating point but not another. For instance, including the "smart pixel" data filtering assumes that the tracking data that are removed do not significantly affect the performance of classifiers within the L1T and HLT systems, which is based on the work carried out by Yoo *et al.* to establish the optimal operating point for this data-filtering technique.[24] Automated optimization of a system with this data reduction could not find the optimal point for how much data to remove, since this high-level model cannot evaluate the validity of the underlying assumption that is based on the physics, experimental setup, and sensors involved in the design.

For these reasons, we promote the use of SystemFlow and other system evaluation tools for use in evaluating systems with intentionally set parameters that can be validated as meaningful and in line with underlying assumptions that can guide future decisions, rather than as a tool for use within automated discovery. However, future tools that could be informed by physics and system design decisions may be able to address this possibility.

VI. CONCLUSION

The amount of data collected by scientific experiments will continue to grow, as will the need to process the data. Simultaneously, the capabilities of computing workflows will become increasingly dependent on the application and integration of novel algorithms and specialized hardware, creating heterogeneous systems. However, the design, validation, and integration of new algorithms and specialized hardware can require significant investment. In order to justify the design of a heterogeneous system, the opportunities and trade-offs it offers over conventional approaches must be rigorously projected.

To address this need, we describe the SystemFlow framework, an approach that allows computational workflows to be modeled using high-level characteristics. This allows for rapid investigation of alternative system configurations using preliminary data and results. We apply this approach to characterize a high-energy physics experiment and to project the impact of integrating GPU processing, improved early-stage classification, and ML-based front-end data reduction into this system. We show that these integrations could provide major reductions in the power required for the system to operate (~60%) but that more advances will be required in order for the system to match or surpass the productivity of earlier experiments running under less challenging experimental conditions.

ACKNOWLEDGMENTS

This work was supported by DOE ASCR and BES Microelectronics Threadwork. This material is based upon work supported by the U.S. Department of Energy, Office of Science, under Contract No. DE-AC02-06CH11357.

The authors thank the following contributors for many fruitful discussions: Jennet Dickinson, Nhan Tran, Jieun Yoo, Jinlong Zhang, Alexander A. Paramov, Kevin Brown, and Tupendra Oli.

AUTHOR DECLARATIONS

Conflict of Interest

The authors have no conflicts to disclose.

Author Contributions

Wilkie Olin-Ammentorp: Conceptualization (equal); Data curation (equal); Software (equal); Visualization (equal); Writing – original draft (equal); Writing – review & editing (equal). **Xingfu Wu**: Methodology (equal); Writing – review & editing (equal). **Andrew A. Chien**: Conceptualization (equal); Funding acquisition (equal); Project administration (equal); Supervision (equal); Writing – review & editing (equal).

DATA AVAILABILITY

The data that support the findings of this study are openly available at https://github.com/wilkieolin/system_flow.

APPENDIX: RESEARCH METHODS

1. Smart pixels training cost

To compare the amount of energy used by training vs inference for the multilayer perceptron (MLP) network used in the "smart pixels" approach, we establish an upper bound on the amount of energy a reasonable implementation would require for training. Using code and data shared by the smart pixels project, we implemented the MLP network in TensorFlow and trained it on the data to reach a similar level of performance. Using a commonly available processor (Mac M1 Max) without leveraging GPU features, this training took under an hour. The worst-case thermal design power (TDP) of the M1 Max and similar laptop processors is ~100 W. Using an upper bound of 1 h for training and this TDP, we estimated an energetic cost of 360 kJ for the network. Given that we estimate the "best-case" inference situation for the smart pixels in which power scales linearly with the number of chips to reach 2.3 kW for the entire inner tracker system, the amount of power used for training is not significant compared with the amount of power used for inference by the classifier over a run of the detector—~12–24 h, based on experimental conditions and how long the beam remains stable within the accelerator.

2. Classifier model

Messages that represent data that should be retained by the data acquisition system (DAQ) and messages that should be discarded represent two distinct populations. Each processing node in the system may implement a classifier that seeks to determine which population a message is a member of, tagging it either to be retained and sent to the next node in the workflow graph or to be discarded.

Various strategies may be used to implement a classifier, seeking to trade off accuracy for speed or processing requirements. In order to avoid the need to reimplement specific details of each system, each is interpreted as fundamentally being capable of producing a "score" for each incoming message. Ideally, messages to be discarded will produce a score of zero, producing a distribution of statistics that represent the "false" samples. Ideally, "true" messages to be retained produce score values that are high and easily distinguishable from scores in the false distribution. The cumulative distribution function (CDF) of each population's scores may be calculated from parametric or empirical distributions,

$$CDF_{\mathbf{Z}}(z) = P(\mathbf{Z} \leq z),$$

where $\mathbf{Z}$ represents scores produced by the classifier on a given population and z represents an individual score.

Each classifier must independently determine whether to pass or reject a message based on the score it produces, requiring a "threshold" to be set. Messages producing scores above this threshold will be accepted; those falling below it will be rejected. The value of this threshold will affect both the amount of data sent by one node in the workflow to the next node and the classifier's performance. The independent variable, determined by the system-level requirements, is the amount of data that the classifier sends to the next processing node; classification performance responds to this parameter.

In order to determine the performance of each classifier, the distribution of scores for the false and true message populations is weighted by the number of incoming true and false messages,

$$CDF_{node}(z) = \frac{n_{true} \cdot CDF_{true}(z) + n_{false} \cdot CDF_{false}(z)}{n_{true} + n_{false}}.$$

The threshold for classification (z_T) will be set at a score that rejects the desired number of samples,

$$z_T = \arg\min\left(\left|\frac{n_{rejected}}{n_{inputs}} - CDF_{node}(z)\right|\right).$$

This determines the proportion of samples from the underlying distribution that will be retained or rejected,

$$TN = n_{false} \cdot CDF_{false}(z_T),$$

$$FP = n_{false} \cdot \left(1 - CDF_{false}(z_T)\right),$$

$$TP = n_{true} \cdot \left(1 - CDF_{true}(z_T)\right),$$

$$FN = n_{true} \cdot CDF_{true}(z_T).$$

The rate of true and false negatives and positives may be used to construct a contingency matrix for the classifier at a given threshold. Messages classified as false are discarded, and messages classified as true are passed to the next classifier.

In order to obtain the contingency matrix for the system as a whole, true negatives and false negatives are summed across the entire system, and positives are produced only by the final output where messages are saved to disk.

3. Level-1 trigger

Accurately capturing a distribution of scores produced by classifiers analyzing messages is key to implementing a model that can faithfully produce system-level performance. For this reason, the Level-1 Trigger (L1T)'s performance is captured in detail to produce these scores.

The phase-1 L1T centers on detecting the existence of certain features—electrons, photons, muons, tau leptons, and hadronic jets—with momenta or energy sums above given thresholds (30 GeV for electrons and photons, 22 GeV for muons, 38 GeV for tau leptons, and a total of 320 GeV for hadronic jets). The L1T does not directly produce scores but accepts any event with one or more of these signatures. To instead produce scores, we examine the efficiency curves of each of these object detections ("trigger paths") and interpret them as a probability that a particle with a given momentum will be correctly detected. To produce a distribution of near-threshold events, an exponential distribution of particle momenta is fitted to reproduce the trigger rates produced by the L1T during the LHC run-2 (2018),

$$rate_{object}(\lambda) = \int_0^\infty efficiency_{object}(p) \cdot PDF_{object}(\lambda, p) \cdot dp,$$

$$PDF_{object}(\lambda, p) = \lambda \cdot \exp\left(-1 \cdot \lambda \cdot x\right) \cdot (x > 0),$$

$$\lambda_{object} = \arg\min\left(\left\|rate_{object}(\lambda) - rate_{object,empirical}\right\|\right).$$

In order to generate a model of scores produced by classifying objects within the HLT, particles are randomly generated from the distribution of object momenta fit to the triggers. Objects that fall below the selection threshold represent the null distribution (samples to be discarded), and objects with momenta above the threshold represent the positive distribution (samples to be retained). By arbitrarily generating many samples ($n = 50\,000$), the distribution of null and positive scores for each L1T object detection trigger is generated. These null and positive distributions are summed to calculate overall L1T scores for each sample and used to produce the confusion matrix of the L1T at a given selection threshold.

4. High-level trigger

The high-level trigger (HLT) contains hundreds of different features (the HLT "menu") that may lead to a sample being retained for offline storage and analysis. Modeling in full each segment of this menu is beyond the scope of this work. Instead, the HLT is approximated by utilizing results from the accepted run-2 (2018) performance of the CMS HLT. Six representative triggering paths for different physics objects (B2G/Exotica, Higgs, SUSY, Muons, Tracking, and Tau) are selected, and the efficiency curves of each path are fit to an exponential distribution of objects in the same manner as the L1T. However, each path is selected independently, and scores are not summed across each trigger but taken one at a time to generate the distribution of null and positive scores applied to generate a confusion matrix for a given selection threshold.

5. Quantifying the cost of classifier errors

Discarding a data sample that should have been retained (a false negative) carries a different energy cost from that when retaining a sample that should have been discarded (false positive). This difference of energy within the data-processing system can be quantified by using the SystemFlow model.

As messages pass through the system, they may be discarded at any processing node. As a message propagates, the amount of energy used to process and transmit it increases. The total energy (TE) utilized to reach any node n in the computational graph $\mathcal{G}$ can be found by traversing the paths available to reach it,

$$TE(\mathcal{G}, n) = E_n + \sum TE(predecessor(\mathcal{G}, n)).$$

The mean energy taken to reach a node is the average of the total energy taken by each of its predecessor nodes, weighted by the number of messages N from each node,

$$MTE(\mathcal{G}, n) = \frac{1}{\sum_p^{predecessors(\mathcal{G},n)} N_p} \cdot \sum_p^{predecessors(\mathcal{G},n)} TE(\mathcal{G}, p).$$

Using the energy taken to reach a given node, we may calculate the differential in power between errors of type I (false positives) and errors of type II (false negatives). These differentials are made in comparison with correct outputs (true positives and true negatives).

In the case of a true positive, a true message reaches the output node (root node of the tree graph). The energy taken is thus the mean total energy (MTE) of the output node,

$$E_{TP} = MTE(\mathcal{G}, root(\mathcal{G})).$$

True negatives may be found not only at the output node but also at any processing node. The amount of energy produced by a true negative result is thus variable, but the expected value over many messages may again be found. The MTE of each classifier node is weighted by the number of messages it discards, producing an average energy consumed by each true negative,

$$E_{TN} = \frac{1}{\sum_c classifiers(\mathcal{G})N_c} \cdot \sum_c classifiers(\mathcal{G})N_c \cdot MTE(\mathcal{G}, c).$$

The cost of false (incorrect) decisions is estimated in comparison with that of true (correct) decisions. In the case of a false positive, the message consumes the cost of a true positive while providing no useful information. Its cost is thus the differential between the amount of energy it consumed during processing and the amount it should have consumed,

$$E_{FP} = E_{TP} - E_{TN}.$$

In the case of a false negative, a message consumes the expected energy of a true negative while valuable information is discarded. In this case, the data processing consumes less energy, but this is countered by the fact that the system must keep running longer to provide true, useful samples. The cost of a false negative is thus estimated as the cost of processing a true negative plus the estimated cost of acquiring a "replacement" true positive. This replacement comes with both the cost of processing a true positive and several false positives, multiplied by the ratio of true negatives to one true positive at the output node,

$$E_{FN} = E_{TN} + \frac{TN(root(\mathcal{G}))}{TP(root(\mathcal{G}))} \cdot E_{TN} + E_{TP}.$$

REFERENCES

[1] M. Hammer, K. Yoshii, and A. Miceli, "Strategies for on-chip digital data compression for X-ray pixel detectors," J. Instrum. **16**, P01025 (2021).

[2] R. Jongerius, S. Wijnholds, R. Nijboer, and H. Corporaal, End-to-end compute model of the square kilometre array.

[3] Paul Laycock for the DUNE Collaboration, "DUNE software and computing challenges," EPJ Web Conf. **251**, 03041 (2021).

[4] P. Das, "An overview of the trigger system at the CMS experiment," Phys. Scr. **97**, 054008 (2022).

[5] CMS Collaboration, "The phase-2 upgrade of the CMS data acquisition and high level trigger," Technical Design Report No. CERN-LHCC-2021-007, 2021.

[6] K. Rupp, 42 Years of microprocessor data, 2018.

[7] More Moore Team, International Roadmap for Devices and Systems, 2022 Update, 2022.

[8] A. Reuther, P. Michaleas, M. Jones, V. Gadepally, S. Samsi, and J. Kepner, "AI accelerator survey and trends," in *2021 IEEE High Performance Extreme Computing Conference (HPEC)* (IEEE, 2021), pp. 1–9, ISBN: 9781665423694; arXiv:2109.08957.

[9] J. L. Hennessy and D. A. Patterson, "A new golden age for computer architecture," Commun. ACM **62**, 48–60 (2019).

[10] CMS Collaboration, S. Chatrchyan, G. Hmayakyan, V. Khachatryan, A. Sirunyan, W. Adam, T. Bauer, T. Bergauer, H. Bergauer, M. Dragicevic *et al.*, "The CMS experiment at the CERN LHC," JINST **3**, S08004 (2008).

[11] M. T. Augustine, "A survey on universal approximation theorems," arXiv:2407.12895 [cs.LG] (2024).

[12] T. Arcomano, I. Szunyogh, A. Wikner, J. Pathak, B. R. Hunt, and E. Ott, "A hybrid approach to atmospheric modeling that combines machine learning with

a physics-based numerical model," J. Adv. Model. Earth Syst. **14**, e2021MS002712 (2022).
[13]K. Choudhary, B. DeCost, C. Chen, A. Jain, F. Tavazza, R. Cohn, C. W. Park, A. Choudhary, A. Agrawal, S. J. L. Billinge, E. Holm, S. P. Ong, and C. Wolverton, "Recent advances and applications of deep learning methods in materials science," npj Comput. Mater. **8**, 59 (2022).
[14]T. Ajayi, D. Blaauw, T.-B. Chan, C.-K. Cheng, V. A. Chhabria, K. Choo, M. Coltella, S. Dobre, R. Dreslinski, M. Fogaca, S. Hashemi, A. Hosny, A. B. Kahng, M. Kim, J. Li, Z. Liang, U. Mallappa, P. Penzes, G. Pradipta, S. Reda, A. Rovinski, K. Samadi, S. S. Sapatnekar, L. Saul, C. Sechen, V. Srinivas, W. Swartz, D. Sylvester, D. Urquhart, L. Wang, M. Woo, and B. Xu, OpenROAD: Toward a self-driving, open-source digital layout implementation tool chain, 2019, https://par.nsf.gov/servlets/purl/10171024.
[15]A. B. Kahng, T. Spyrou, and E. Depts, The OpenROAD project: Unleashing hardware innovation, 2019, https://vlsicad.ucsd.edu/Publications/Conferences/383/c383.pdf.
[16]S. Cherry, "Edholm's law of bandwidth," IEEE Spectrum **41**, 58–60 (2004).
[17]A. Biswas, "Universal Chiplet Interconnect Express (UCIe)®: An open standard for developing a successful chiplet ecosystem," in *2021 IEEE Hot Chips 33 Symposium (HCS)* (IEEE, Palo Alto, CA, 2021), pp. 1–22.
[18]B. Pezeshki, R. Kalman, A. Tselikov, and C. Danesh, "High speed light microLEDs for visible wavelength data communication," in *Light-Emitting Devices, Materials, and Applications XXV*, edited by M. Strassburg, J. K. Kim, and M. R. Krames (SPIE, 2021), p. 20.
[19]M. Wade, E. Anderson, S. Ardalan, P. Bhargava, S. Buchbinder, M. L Davenport, J. Fini, H. Lu, C. Li, R. Meade, C. Ramamurthy, M. Rust, F. Sedgwick, V. Stojanovic, D. Van Orden, C. Zhang, C. Sun, S. Y. Shumarayev, C. O'Keeffe, T. T. Hoang, D. Kehlet, R. V. Mahajan, M. T. Guzy, A. Chan, and T. Tran, "TeraPHY: A chiplet technology for low-power, high-bandwidth in-package optical I/O," IEEE Micro **40**, 63–71 (2020).
[20]ATLAS Collaboration, Reconstruction wall time per event on the average number of interactions per bunch crossing, 2023, https://twiki.cern.ch/twiki/bin/view/AtlasPublic/ComputingandSoftwarePublicResults.
[21]A. Sirunyan *et al.*, "Particle-flow reconstruction and global event description with the CMS detector," J. Instrum. **12**, P10003 (2017); arXiv:1706.04965 [physics.ins-det].
[22]J. Stirling, "Parton luminosity and cross-section charts," private communication (2023).
[23]V. Khachatryan *et al.*, "The CMS trigger system," J. Instrum. **12**, P01020 (2017); arXiv:1609.02366 [physics.ins-det].
[24]J. Yoo, J. Dickinson, M. Swartz, G. Di Guglielmo, A. Bean, D. Berry, M. Blanco Valentin, K. DiPetrillo, F. Fahim, L. Gray, J. Hirschauer, S. R. Kulkarni, R. Lipton, P. Maksimovic, C. Mills, M. S. Neubauer, B. Parpillon, G. Pradhan, C. Syal, N. Tran *et al.*, "Smart pixel sensors: Towards on-sensor filtering of pixel clusters with deep learning," Mach. Learn.: Sci. Technol. **5**, 035047 (2024).
[25]E. A. Moreno, O. Cerri, J. M. Duarte, H. B. Newman, T. Q. Nguyen, A. Periwal, M. Pierini, A. Serikova, M. Spiropulu, and J. R. Vlimant, "JEDI-net: A jet identification algorithm based on interaction networks," Eur. Phys. J. C **80**, 58 (2020); arXiv:1908.05318.
[26]ATLAS Collaboration, "Search for new phenomena in two-body invariant mass distributions using unsupervised machine learning for anomaly detection at $\sqrt{s}$ = 13 TeV with the ATLAS detector," arXiv:2307.01612 [hep-ex] (2023).
[27]CMS Collaboration, "The phase-2 upgrade of the CMS tracker technical design report," Technical Report No. CERN-LHCC-2017-009, 2017.
[28]CMS Collaboration, "The phase-2 upgrade of the CMS level-1 trigger," Technical Design Report No. CERN-LHCC-2020-004, 2020.
[29]Commercial QSFP Gigabit Ethernet Module Power Usage, 2023.
[30]S. Amrouche, L. Basara, P. Calafiura, V. Estrade, S. Farrell, D. R. Ferreira, L. Finnie, N. Finnie, C. Germain, V. V. Gligorov, T. Golling, S. Gorbunov, H. Gray, I. Guyon, M. Hushchyn, V. Innocente, M. Kiehn, E. Moyse, J.-F. Puget, Y. Reina, D. Rousseau, A. Salzburger, A. Ustyuzhanin, J.-R. Vlimant, J. S. Wind, T. Xylouris, and Y. Yilmaz, "The tracking machine learning challenge: Accuracy phase," arXiv:1904.06778 (2019).
[31]B. Dally, "Hardware for deep learning," in *2023 IEEE Hot Chips 35 Symposium (HCS)* (IEEE Computer Society, 2023), pp. 1–58.
[32]M. Feickert and B. Nachman, "A living review of machine learning for particle physics," arXiv:2102.02770 [hep-ph] (2021).
[33]S. Mondal and L. Mastrolorenzo, "Machine learning in high energy physics: A review of heavy-flavor jet tagging at the LHC," Eur. Phys. J.: Spec. Top. **233**, 2657 (2024); arXiv:2404.01071 [hep-ex].
[34]J. Bardhan, T. Mandal, S. Mitra, C. Neeraj, and M. Patra, "Unsupervised and lightly supervised learning in particle physics," Eur. Phys. J.: Spec. Top. **233**, 2559 (2024); arXiv:2403.13676 [hep-ph].
[35]A. Bogatskiy, T. Hoffman, and J. T. Offermann, "19 parameters is all you need: Tiny neural networks for particle physics," arXiv:2310.16121 [hep-ph] (2023).